VYSOKÉ UČENÍ TECHNICKÉ V BRNĚ
BRNO UNIVERSITY OF TECHNOLOGY

FAKULTA INFORMAČNÍCH TECHNOLOGIÍ
ÚSTAV INFORMAČNÍCH SYSTÉMŮ

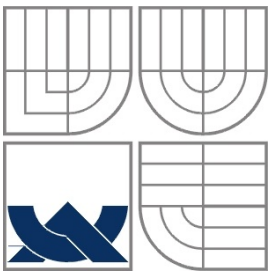
FACULTY OF INFORMATION TECHNOLOGY
DEPARTMENT OF INFORMATION SYSTEMS

DETEKCE ÚTOKU POMOCÍ ANALÝZY SYSTÉMOVÝCH LOGŮ

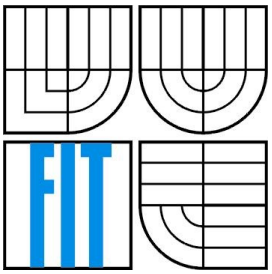
DIPLOMOVÁ PRÁCE
MASTER'S THESIS

AUTOR PRÁCE Bc. ONDŘEJ HOLUB
AUTHOR

BRNO 2010



VYSOKÉ UČENÍ TECHNICKÉ V BRNĚ
BRNO UNIVERSITY OF TECHNOLOGY



FAKULTA INFORMAČNÍCH TECHNOLOGIÍ
ÚSTAV INFORMAČNÍCH SYSTÉMŮ

FACULTY OF INFORMATION TECHNOLOGY
DEPARTMENT OF INFORMATION SYSTEMS

DETEKCE ÚTOKU POMOCÍ ANALÝZY SYSTÉMOVÝCH LOGŮ

ATTACK DETECTION BY ANALYSIS OF THE SYSTEM'S LOGS

DIPLOMOVÁ PRÁCE
MASTER'S THESIS

AUTOR PRÁCE
AUTHOR

Bc. ONDŘEJ HOLUB

VEDOUCÍ PRÁCE
SUPERVISOR

Ing. JAN KAŠTIL

BRNO 2010

Zadání diplomové práce/10425/2009/xholub22

Vysoké učení technické v Brně - Fakulta informačních technologií

Ústav počítačových systémů Akademický rok 2009/2010

Zadání diplomové práce

Řešitel: **Holub Ondřej, Bc.**
Obor: Informační systémy
Téma: **Detekce útoku pomocí analýzy systémových logů**
Attack Detection by Analysis of the System's Logs
Kategorie: Bezpečnost

Pokyny:

1. Seznamte se s HIDS (Host Intrusion Detection Systems).
2. Seznamte se s metodami pro detekci anomálií v datech.
3. Navrhněte seznam vlastností, které je třeba zaznamenávat pro úspěšnou detekci útoků.
4. Navrhněte protokol pro přenos zaznamenaných charakteristik systému na vzdálenou stanic.
5. Zaznamenejte charakteristiky napadených i nenapadených počítačů v rámci různých režimů provozu.
6. Navrhněte a implementujte jednotku pro rozpoznání útoků ze sesbíraných dat.
7. Diskutujte dosažené výsledky a navrhněte možná pokračování projektu.

Literatura:

- Dle pokynů vedoucího

Při obhajobě semestrální části diplomového projektu je požadováno:

- Body 1 až 3.

Podrobné závazné pokyny pro vypracování diplomové práce naleznete na adrese http://www.fit.vutbr.cz/info/szz/

Technická zpráva diplomové práce musí obsahovat formulaci cíle, charakteristiku současného stavu, teoretická a odborná východiska řešených problémů a specifikaci etap, které byly vyřešeny v rámci ročníkového a semestrálního projektu (30 až 40% celkového rozsahu technické zprávy).

Student odevzdá v jednom výtisku technickou zprávu a v elektronické podobě zdrojový text technické zprávy, úplnou programovou dokumentaci a zdrojové texty programů. Informace v elektronické podobě budou uloženy na standardním nepřepisovatelném paměťovém médiu (CD-R, DVD-R, apod.), které bude vloženo do písemné zprávy tak, aby nemohlo dojít k jeho ztrátě při běžné manipulaci.

Vedoucí: **Kaštil Jan, Ing.**, UPSY FIT VUT
Datum zadání: 21. září 2009
Datum odevzdání: 26. května 2010

VYSOKÉ UČENÍ TECHNICKÉ V BRNĚ
Fakulta informačních technologií
Ústav počítačových systémů a sítí
612 66 Brno, Božetěchova 2

L.S.

doc. Ing. Zdeněk Kotásek, CSc.
vedoucí ústavu

Abstrakt

Práce pojednává o možnostech detekce útoků a nestandardního chování. Zabývá se problematikou detekčních systémů IDS, jejich klasifikací a metodami, které tyto systémy k detekci útoků využívají. Část práce je věnována seznámení se s existujícími IDS systémy a s vlastnostmi, které jsou nezbytné pro úspěšnou detekci útoků. Další části přibližují metody získávání informací z operačních systémů Microsoft Windows a teoretické metody detekce anomálií v datech. V praktické části se pak práce zaměřuje na návrh a implementaci HIDS aplikace. Výsledná aplikace a její detekční schopnosti jsou ke konci praktické části testovány na několika modelových situacích. Závěr práce tvoří shrnutí získaných poznatků a nastínění směru dalšího vývoje.

Abstract

The thesis deals with the attack detection possibilities and the nonstandard behaviour. It focuses on problems with the IDS detection systems, the subsequent classification and methods which are being used for the attack detection. One part of the thesis presents the existing IDS systems and their properties which are necessary for the successful attack detection. Other parts describe methods to obtain information from the operating systems Microsoft Windows and it also analyses the theoretical methods of data abnormalities. The practical part focuses on the design and implementation of the HIDS application. The final application and its detection abilities are tested at the end of the practical part with the help of some model situations. In the conclusion, the thesis sums up the gained information and shows a possible way of the future development.

Klíčová slova

IDS, HIDS, systémy detekce narušení, detekce útoků, monitorování, detekce anomálií, Microsoft Windows, WMI, Windows API, neuronové sítě, Kohonenovy samoorganizační mapy.

Keywords

IDS, HIDS, intrusion detection systems, attack detection, monitoring, anomaly detection, Microsoft Windows, WMI, Windows API, neural networks, Kohonen’s self-organizing maps.

Citace

Holub Ondřej: Detekce útoku pomocí analýzy systémových logů, diplomová práce, Brno, FIT VUT v Brně, 2010.

Detekce útoku pomocí analýzy systémových logů

Prohlášení

Prohlašuji, že jsem tuto diplomovou práci vypracoval samostatně pod vedením Ing. Jana Kaštila.

Uvedl jsem všechny literární prameny a publikace, ze kterých jsem čerpal.

……
Ondřej Holub
26. 5. 2010

Poděkování

Děkuji všem, kteří mě při práci podporovali, zvláště pak svému vedoucímu práce panu Ing. Janu Kaštilovi.

Obsah

Úvod

S rozvojem informačních technologií se stala bezpečnost jednou z nejdůležitějších oblastí informatiky. S rychlým vývojem počítačů a jejich softwarového vybavení se velmi rychle rozšiřují bezpečnostní hrozby a rizika, kterým je nutno čelit. Jedním ze způsobů obrany je detekce nežádoucích aktivit, na jejímž základě pak mohou být podniknuty nezbytné kroky k zamezení obdobným situacím v budoucnu.

Cílem této práce je seznámení s možnostmi systémů detekce narušení, přiblížení principů činnosti těchto systémů, studium metod, které tyto systémy používají a v konečné fázi návrh a implementace detekčního systému HIDS. Práce se rovněž zabývá zhodnocením přínosu IDS v oblasti bezpečnosti počítačových systémů. Je členěna do několika kapitol, které se podrobněji věnují nastíněnému tématu.

Práce navazuje na semestrální projekt, ze kterého vychází její teoretická část. Některé části, jako například skupiny vlastností používaných při detekci anomálií, byly upraveny na základě zkušeností, získaných při návrhu a implementaci praktické části práce.

První kapitola je teoretickým úvodem do problematiky systémů detekce proniknutí. Její první část je věnována definici těchto systémů a jimi prováděných činností, zabývá se jejich obecnou architekturou. Následuje část, jež se detailně zabývá různými aspekty detekčních systémů a následně je hierarchicky rozděluje do několika skupin. Pro jednotlivé kategorie detekčních systémů jsou diskutovány jejich kladné a záporné vlastnosti.

Druhá kapitola je zaměřena na studium existujících systémů detekce proniknutí, zabývá se funkcemi, které tyto systémy nabízejí a věnuje pozornost jejich přednostem i nedostatkům.

Třetí kapitola se zaměřuje na problematiku detekce proniknutí, útoků a škodlivých aktivit pomocí HIDS systémů, které jsou podkategorií IDS systémů. Text se velmi podrobně věnuje jednotlivým skupinám vlastností, které jsou nutné pro úspěšnou detekci útoků detekčními systémy HIDS, které jsou podkategorií IDS systémů.

Čtvrtá kapitola se zabývá z teoretického hlediska systémovými funkcemi, které slouží k monitorování systémových prostředků a k získávání informací o aktivitách, odehrávajících se v rámci operačních systémů Microsoft Windows. Její první část se věnuje rozhraní Windows API, jež nabízí operační systém, jakož i možnostem jeho využití pro přístup a k získávání statistických dat pro monitorování systému. Jsou zde nastíněny některé velmi často volané funkce a problémy související s jejich používáním. V druhé části této kapitoly se práce zaobírá oblastí služeb systémů Microsoft Windows, které mají za úkol zpracovávat a následně pak prezentovat data o systémových prostředcích. Jsou zde zmíněny způsoby přístupu k těmto datům a možné problémy související s těmito službami.

Pátá kapitola si bere za cíl přiblížit metody detekce anomálií v datech. První část kapitoly se zabývá teoretickým úvodem detekce anomálií v datech. V dalších částech jsou popsány jednotlivé přístupy a metody, pomocí kterých je možno detekovat anomálie v datech a tím i zaznamenat abnormální chování pro účely HIDS detekčních systémů.

Šestá kapitola je stěžejní kapitolou celé práce. Na základě shromážděných informací se věnuje návrhu a následné implementaci detekční HIDS aplikace, pracující na bázi detekce anomálií v datech. V této kapitole jsou detailně popsány nejdůležitější části aplikace a je prezentována celá řada testů, odpovídajících reálným podmínkám, které mohou při provozu aplikace nastat.

V sedmé závěrečné kapitole této práce jsou shrnuty poznatky získané při studiu problematiky IDS. Jsou zhodnoceny dosažené výsledky a je nastíněn směr dalšího možného vývoje.

1 Systémy detekce proniknutí

Systémy detekce proniknutí se v cizojazyčné literatuře nazývají Intrusion Detection Systems, zkráceně jsou pak označovány jako IDS. Jejich primárním účelem je detekce útoků a odhalování hrozeb, které systému bezprostředně hrozí. Existují příbuzné systémy nazývané Intrusion Prevention Systems, zkráceně IPS, které mají za úkol útokům předcházet, pokud je to možné.

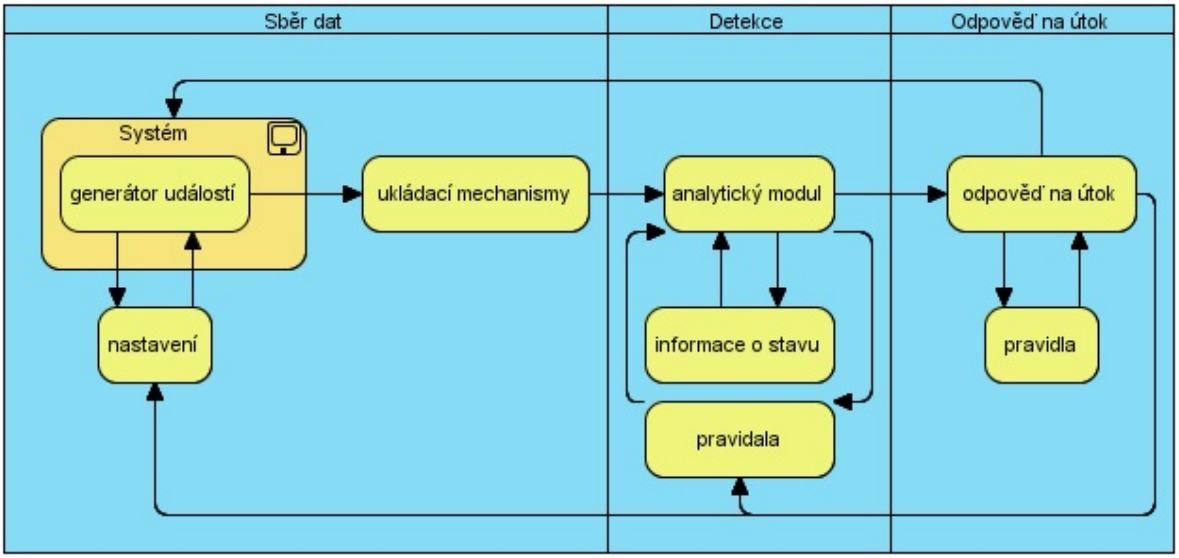
Hlavním posláním IDS je detekce nepřátelských aktivit v síti a v prostředí operačních systémů monitorovaných výpočetních prostředků. Mezi důležité vlastnosti systémů pro odhalení průniků patří jejich schopnost odhalovat jak úspěšně, tak i neúspěšně zakončené nepřátelské aktivity. Navíc jsou schopny ve velkém množství případů detekovat i akce, které vlastním útokům předcházejí. Samotný útok je velmi často předcházen abnormálním chováním a podezřelými aktivitami ať už v prostředí sítě nebo v rámci operačního systému. Mezi takové neobvyklé činnosti se v síťovém prostředí řadí například vznik nezvyklého množství síťových spojení, skenování síťových portů a další ne zcela obvyklé jevy. V prostředí operačního systému mohou přípravu nebo probíhající útok naznačovat například neobvyklé změny v systémových souborech nebo registru, abnormální využití systémových prostředků nebo zvýšená komunikace pomocí zasílání zpráv mezi aplikacemi v systému. Při zjištění nezvyklé aktivity systémy IDS informují o této skutečnosti administrátora, který by měl být schopen s podporou zaznamenaných informací o hrozícím nebezpečí situaci vyhodnotit, zareagovat a vykonat takové kroky, které povedou k zamezení útoku a k případné eliminaci detekované hrozby v budoucnosti. Ochranu proti takovýmto útokům nelze zabezpečit pomocí běžných hardwarových nebo softwarových prostředků, nejsou zcela účinné ani šifrovací systémy nebo síťové monitorovací nástroje, sloužící například k odhalování DoS útoků. Tento fakt dává prostor existenci IDS a IPS, které se v této oblasti s ostatními bezpečnostními nástroji nepřekrývají z hlediska nabízené funkcionality a schopnostmi detekce, ale stávající nástroje a prostředky doplňují a pomáhají tak vzniku komplexnějšího a účinnějšího systému ochrany.

1.1 Architektura systémů IDS

V oblasti současných systémů detekce průniků existuje celá řada koncepcí, které jsou od sebe navzájem velmi odlišné a jsou uzpůsobeny k nasazení v různých oblastech a v rozličných typech systémů.

V praxi se používají jednovrstvé architektury, kde jediná komponenta plní všechny funkce. Existují i vícevrstvé architektury s hierarchickým uspořádáním, které využívají senzory, agenty a manažera. Posledním běžně používaným typem architektury je Peer-to-Peer architektura, která je tvořena rovnocennými spolupracujícími komponentami.

Nejobecnější teoretický model IDS systému by měl obsahovat alespoň moduly generátoru událostí, analýzy a protiopatření a měl by disponovat ukládacími mechanismy.



Obrázek 1.1: Model architektury IDS [1]

Prvním nezbytným modulem je **generátor událostí**, jež má za úkol šíření informací o událostech vzniklých v systému. Generátor událostí je tvořen čidlem, které detekuje probíhající události v systému bez ohledu na to, zda se jedná o útoky či běžnou aktivitu a předává je dalším modulům IDS ke zpracování a k analýze. Data jsou uložena ve skladišti logů a ihned nebo později zpracována analytickým modulem [1].

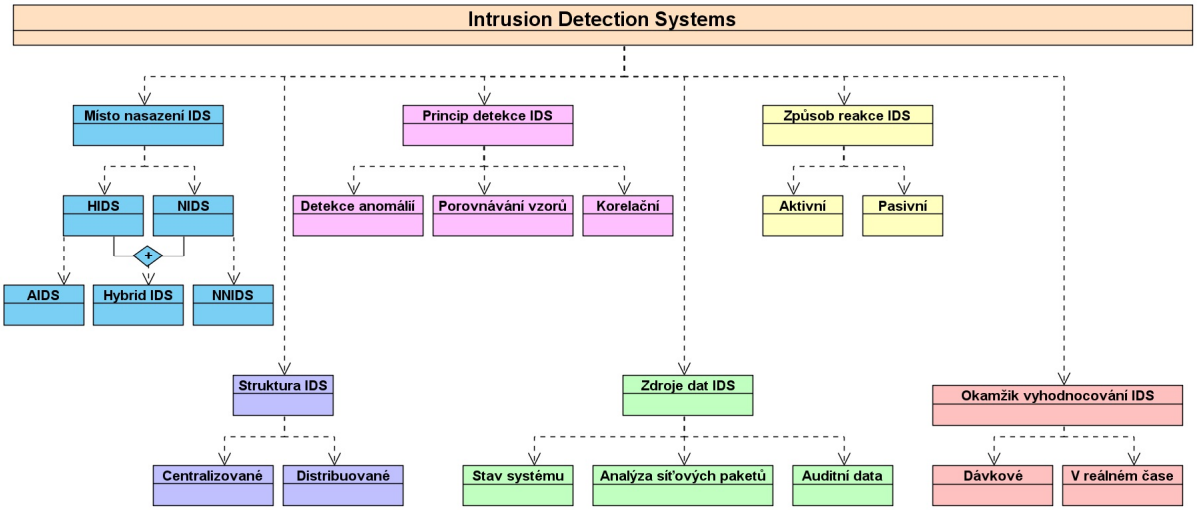
Analytického modul je pro úspěšnost celé detekce naprosto klíčovým prvkem. V tomto modulu je implementován detekční algoritmus, závislý na konkrétním návrhu IDS systému. Na jeho efektivitě závisí schopnost systému rozpoznat útoky od běžné neškodné aktivity. Tento modul je nejsložitějším prvkem systému a existuje řada metod, kterými se data analyzují. Jedná se o metody, které mohou být například zaměřeny na statistickou detekci anomálií, na detekci pomocí porovnávání s databází uložených vzorů známých útoků nebo o metody, založené na pokusech napodobit chování podobné lidskému imunitnímu systém [1].

Protože generátory událostí a analytické moduly produkují značné množství dat užitečných pro správce systému a pro další zpracování IDS systémy, je nutné tato data ve vhodné podobě uchovávat a poskytovat povolaným osobám. K tomuto účelu je v systému modul zvaný **ukládací mechanismy**. Neuchovává zpravidla úplné zápisy, ale většinou jen charakteristiku zjištěných útoků, která postačuje k jejich identifikaci a popisuje jejich chování.

Poslední modul může být implementován rozdílně v různých IDS. Jedná se o **modul odpovědi** na útok, který má ve většině systémů za úkol pouze zasílat výstrahy při zjištění abnormálního chování nebo při detekci útoku. V propracovanějších systémech mohou tyto moduly vykonávat akce, kterými se systém snaží zabránit probíhajícímu útoku nebo zamezit pokusům o podobný útok v budoucnu.

1.2 Rozdělení systémů IDS

Systémy pro detekci průniků je možno rozdělit dle několika hledisek. Nejobecnější dělení IDS je dle místa jejich nasazení. Toto má vliv na kategorii vstupních parametrů, pomocí nichž detekují potenciálně nebezpečné situace a útoky. V závislosti na lokalizaci IDS se odvíjí jejich další dělení, které je podrobně popsáno v následujícím textu.



Obrázek 1.2: Dělení IDS

1.2.1 Místa nasazení IDS

Podle místa nasazení lze rozlišit v obecné rovině 5 druhů systémů. Prvním typem jsou takzvané HIDS, neboli Host Based Intrusion Detection Systems, které jsou lokalizovány přímo na jednotlivých stanicích a serverech, kde detekují útoky a nebezpečné aktivity v prostředí jejich operačního systému. Druhou skupinou jsou NIDS neboli Network Based Intrusion Detection Systems, které bývají lokalizovány v jednotlivých segmentech sítě a jejich primárním účelem je monitoring a detekce útoků v oblasti síťového provozu. Třetím typem IDS jsou systémy, které kombinují oba zmíněné přístupy, jsou nejkomplexnější a zároveň nejsložitější. Takové systémy nesou označení Hybrid IDS. Existují ještě systémy NNIDS a AIDS, které jsou podskupinou systémů NIDS a HIDS, blíže se zaměřují na určité aspekty sledování a jsou detailněji popsány v následujících odstavcích.

1.2.1.1 Host Based IDS

HIDS jsou klasickou kategorií detekčních systémů, které používají jako zdroj informací záznamy o chodu sledovaného výpočetního prostředku. Tyto záznamy jsou buď produkovány přímo systémem a aplikacemi nebo případně může být v systému přítomen specializovaný agent, který má za úkol informace ze systému aktivně shromažďovat. Systémy HIDS bývají založeny buď na detekci abnormálního chování uživatele a aplikací nebo na detekci na základě porovnávání vzorů [2].

První kategorie se vyznačuje detekcí aktivit, které nejsou pro daného uživatele typické a neodpovídají jeho běžnému vzoru chování. Ačkoliv se jedná o činnosti odpovídající uživatelově úrovni práv, mohou poukazovat například na zneužití výpočetních prostředků nebo na vykonávání jiných nežádoucích aktivit. Tyto metody jsou založené na detekci anomálií a využívají vzory chování. Vzor typického chování uživatele je možno vytvářet automaticky na základě sledování činnosti uživatele a systému, dále je nutno získané vzory postoupně v čase přizpůsobovat a adaptivně měnit. Vyhodnocování útoků se pak často realizuje metodou prahování, kdy je předem určena stupnice rizikovosti chování a podle zjištěného stupně je rozhodnuto, zda se jedná o útok či nikoliv.

Druhý přístup se zabývá analýzou jednoznačných indikátorů činnosti, která vykazuje chování vymykající se standardnímu chování. Jedná se zejména o takové chování, které nenastává při běžné manipulaci s výpočetním prostředkem. Jsou sledovány především pokusy o změnu systémových souborů a registrů systému nebo pokusy o zkoumání vnitřní struktury a chodu systému. Tyto metody bývají založeny na porovnávání aktuálního chování a nastalých událostí s databází obsahující vzory škodlivých činností. Pro úspěšnou detekci je nezbytné, aby IDS disponoval jejich aktuální databází.

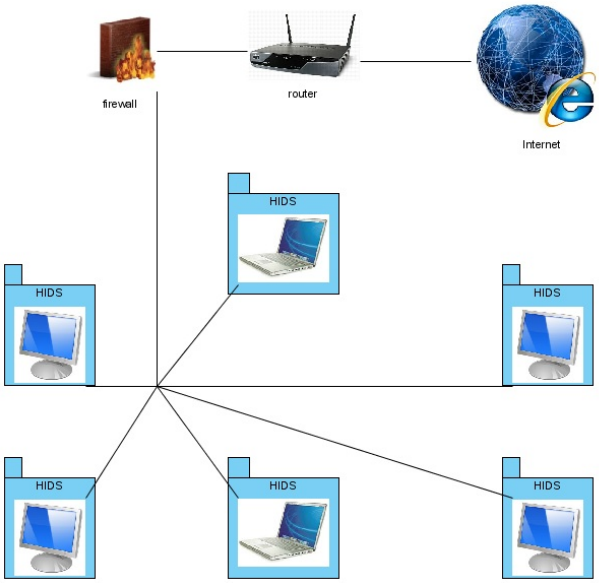
Na kvalitě vzorů chování a vzorků nebezpečných činností závisí z velké části schopnost a úspěšnost detekce útoků.

HIDS bývají do systému velmi těsně integrovány a jsou velmi výkonnými nástroji pro monitorování útoků na aplikační vrstvě. K získání informací pro analýzu se zaměřují HIDS na pět základních oblastí [3][2]:

- Analýza známých signatur
  - Sledování síťového provozu a analýza na úrovni paketů.
  - Detekce známých nebezpečných procesů a služeb.
  - Detekce síťových útoků na základě netypické konfigurace síťových portů používaných ke komunikaci.
- Analýza chování
  - Sledování příchozích síťových spojení. HIDS pak podrobují analýze všechna příchozí a odchozí spojení, čímž dokáže systém odhalovat abnormální aktivity typu skenování portů či nezvyklé počty komunikačních spojení.
  - Monitoring akcí prováděných uživateli s oprávněním vysoké úrovně. Bývá analyzována četnost akcí, vyžadujících vysoké oprávnění.
- Analýza logů
  - Sledování pokusů o přihlášení nebo odhlášení se od systému. Systémem detekce průniků je monitorována četnost přihlašování, opakovaná neúspěšná přihlášení, frekvence pokusů o přihlášení k systému nebo lokality, ze kterých jsou pokusy o přístup do systému iniciovány.
  - Atypické uspořádání činností při běhu aplikace nebo operačního systému, které může implikovat změnu jejich chování.
  - Sledování průniků do jádra systému, kdy tato oblast je často sledována v operačních systémech na bázi Unixu.
- Analýza systémových volání
  - Monitoring meziprocesové komunikace.
  - Analýza příkazů zadávaných uživateli prostřednictvím příkazové řádky.
  - Sledování chyb způsobených přetečením bufferu.
- Sledování integrity souborů a systému
  - Systémy detekce průniků se zaměřují na sledování stavu systémového registru za předpokladu, že je v monitorovaném systému implementován. Předmětem zkoumání bývají změny prováděné v registru a přítomnost podezřelých objektů v kritických místech registru, díky kterým může docházet například ke skrytému spouštění nežádoucích aplikací na pozadí systému. V případě modifikace registru hrozí nefunkčnost systému nebo jeho částí a může být narušena bezpečnost.
  - Sledování integrity systému, kdy jsou systémem detekce monitorovány změny systémových a konfiguračních souborů, protože každá jejich změna může způsobit závažné bezpečnostní problémy nebo vyřadit celý systém z provozu.

Sledování bývá realizováno pomocí kontrolních součtů a snímků stavu systému.

Vzhledem ke své specifičnosti a subjektivnímu hodnocení chování nemohou být tyto systémy nikdy zcela úspěšné, mohou způsobovat řadu falešně pozitivních poplachů, ale na druhou stranu mohou být velkým přínosem k celkové systémové bezpečnosti, neboť jsou schopny detekovat i neznámé a dosud neobjevené útoky a rizikové chování.



Obrázek 1.3: Nasazení HIDS

Pozitivní vlastnosti	Negativní vlastnosti
Detekce útoků neodhalitelných běžnými bezpečnostními prostředky	Sběr informací na každém klientovi a jejich přeposílání centrální konzole může snížit výkon systému a zahltit síť
Detekce útoků vedených skrze šifrovaný síťový kanál	Zkušený útočník může vyřadit HIDS z provozu a být tak nedetekovatelný
Detailní sledování a schopnost identifikace škodlivých procesů	HIDS mohou být vyřazeny útoky DoS, neboť při zahlcení není možno předávat informace o probíhajícím útoku centrální konzole
Identifikace uživatelů podílejících se na škodlivé činnosti	Náročné na systémové prostředky
Použití centrální konzoly k získání celkového pohledu na monitorované systémy (celkový přehled, i když jednotlivé HIDS jsou autonomní)	
Nejsou potíže s přepínanými sítěmi jako u NIDS	

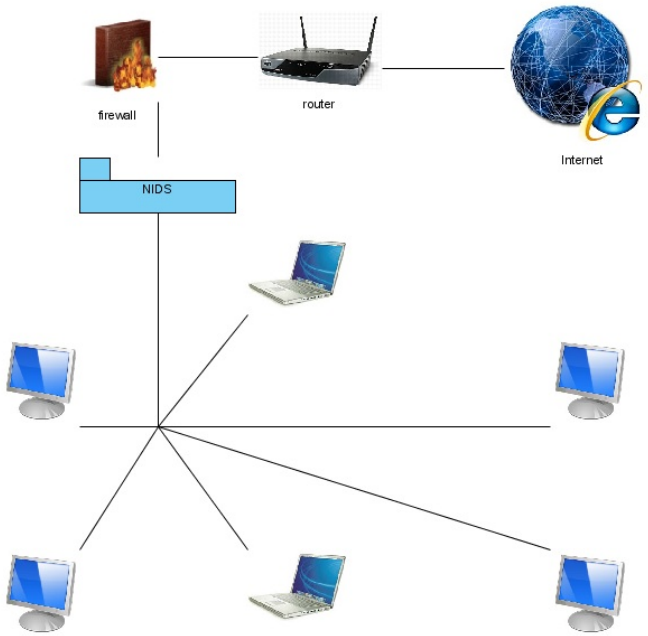
Tabulka 1.1: Výhody a nevýhody HIDS [4]

1.2.1.2 Network Based IDS

NIDS, neboli síťové systémy detekce průniku, jsou obvykle umisťovány do míst, kde dochází ke koncentraci síťového provozu a mají tak možnost analýzy co nejširšího záběru síťové komunikace. Typicky jsou nasazovány na páteřní sítě nebo lokalizovány v přístupových bodech.

NIDS většinou bývají distribuované a skládají se z více částí, rozmístěných v rámci monitorované sítě. Na monitorované body sítě jsou nasazováni tzv. agenti, kteří sledují svěřený úsek sítě. Tito agenti pracují většinou v promiskuitním režimu, což jim umožňuje při monitorování postihnout celé spektrum síťového provozu v daném segmentu sítě bez ohledu na příjemce přenášených dat. Pro jejich činnost není vyžadován aktivní režim a tak nijak neovlivňují přenášená data nebo chování sítě. Pomocí vlastní sítě přenášejí zjištěné údaje k dalším komponentám detekčního systému, jimiž bývají manažer a řídící konzola, která má často za úkol zpracování a analýzu naměřených dat. Konzola data vyhodnotí a v případě útoku informuje o nastalé situaci odpovědného správce [3].

NIDS jsou vhodné pro detekci síťových útoků probíhajících na nižších vrstvách architektury. Je možno detekovat jak útoky zaměřené proti službám běžícím na výpočetním prostředku v síti, tak i distribuované útoky typu DoS či DDoS.



Obrázek 1.4: Nasazení NIDS

Pozitivní vlastnosti	Negativní vlastnosti
Mohou monitorovat celou síť nebo síťový segment pomocí jednoho nebo několika málo vhodně umístěných senzorů	NIDS musí monitorovat kompletní síťový provoz v daném segmentu a při velkém zahlcení sítě nemusí být schopny zpracovat veškeré pakety
Většinou pasivní sledování bez narušení síťového provozu	Špatná funkčnost v přepínaných vysokorychlostních sítích
Snadno zabezpečitelné proti útoku	Nutnost dohledu administrátorů k zajištění efektivity
Nedetekovatelné pro potencionální útočníky	Nelze jednoznačně určit, zda byl útok úspěšně dokončen
Snadná integrace do stávajících sítí	

Tabulka 1.2: Výhody a nevýhody NIDS [4]

1.2.1.3 Hybrid Based IDS

Označením Hybrid IDS jsou míněny hybridní systémy detekce průniku. Jedná se o kombinaci systémů HIDS a NIDS, která přináší jak jejich výhody, tak i nevýhody. Hybridní systémy jsou nejnáročnější na administraci a jsou nejkomplikovanější, zato ovšem přinášejí největší komplexnost a umožňují detekci v oblasti jak síťového provozu, tak i v prostředí operačních systémů jednotlivých výpočetních prostředků.

Hybridní IDS vznikají často jako reakce na současnou situaci v oblasti sítí. Moderní přepínané sítě jsou často problematické pro nasazení klasických NIDS, protože hardware sítí není schopen plné funkčnosti v promiskuitním režimu. Další komplikaci při nasazení klasických NIDS může představovat vysoká rychlost síťové komunikace a fakt, že by mnoho z paketů procházejících detekčním systémem bylo zahozeno. Řešení těchto problémů vede k nasazení hybridních detekčních systémů, díky nimž je možno například detekovat útoky jen v určitých významných uzlech, ve kterých jsou lokalizovány důležité servery, namísto sledování veškerého provozu v rámci celé sítě [5].

Principem činnosti hybridních IDS je používání různých senzorů a jejich synchronizace s centrálním prvkem, který archivuje a vyhodnocuje shromážděné informace. Tento přístup umožňuje detekci celého spektra anomálií a útoků. Většina komerčně nabízených systému spadá právě do kategorie hybridních IDS a využívají výhod HIDS a NIDS.

Pozitivní vlastnosti	Negativní vlastnosti
Výhody HIDS a NIDS	Obtížnější správa a integrace do stávajícího systému
Možnost nasazení detekce jen na vybrané elementy v síti	
Komplexnější ochrana než v případě samostatného HIDS nebo NIDS	
Funkční i v přepínaných a vysokorychlostních sítích	

Tabulka 1.3: Výhody a nevýhody Hybridních IDS [5]

1.2.1.4 Application Based IDS

Aplikační detekční systémy průniku AIDS jsou speciálním případem HIDS. Jejich hlavním účelem je analýza událostí v rámci činnosti aplikací, k čemuž používají ve většině případů data shromažďovaná v transakčních logovacích souborech. Předmětem jejich zkoumání je také interakce mezi uživatelem a sledovanou aplikací. Díky přímému propojení s monitorovanými aplikacemi jsou tyto systémy schopny detekovat podezřelé chování ať už přímo sledované aplikace, tak i uživatelů, kteří se pokoušejí provádět činnosti překračující jim přidělenou úroveň oprávnění. Vzhledem ke své lokaci nemají aplikační detekční systémy problém s šifrovanými daty, neboť na koncových bodech, které monitorují, se nacházejí data v otevřené podobě. Aplikační IDS bývají často tvořeny sítí agentů sbírajících data, kteří je následně zasílají centrálnímu prvku k analýze. Tímto způsobem je možno například pokrýt větší množství stanic v rámci místní sítě. Využití aplikačních IDS spočívá ve sledování kritických aplikací, přičemž bývají často provozovány spolu s HIDS a NIDS systémy k zajištění komplexní bezpečnosti [6].

Pozitivní vlastnosti	Negativní vlastnosti
Snadné na nasazení a administraci	Je možno aplikovat na aplikace, které zaznamenávají svoji činnost
Není problém s šifrovanými daty	Analýza na vysoké úrovni, útoky na nižších úrovních mohou být nepozorovány (např. trojský kůň)
Detailní monitorování aktivit konkrétních uživatelů a aplikací a jejich vzájemné interakce	Nutnost chránit auditní záznamy aplikací

Tabulka 1.4: Výhody a nevýhody Aplikačních IDS [4]

1.2.1.5 Network Node Based IDS

Specifickou skupinu detekčních systémů tvoří takzvané Network Node IDS neboli NNIDS. Vznikly kvůli problematickému nasazení klasických NIDS na moderních vysokorychlostních přepínaných sítích, ve kterých je klasické řešení nespolehlivé a způsobuje velkou ztrátu výkonnosti systému a ztráty paketů. NNIDS jsou schopné vyrovnat se i s nemožností běhu síťového hardware v promiskuitním režimu. Na rozdíl od klasických NIDS, které analyzují veškerý provoz v rámci sítě, sledují NNIDS pouze pakety směřující do jejich uzlu, což značně redukuje jejich počet a snižuje zátěž IDS systému [9]. NNIDS delegují funkčnost IDS až na úroveň jednotlivých výpočetních prostředků v síti, čímž eliminují problémy spojené s přepínáním i vysokou rychlostí komunikace.

Pozitivní vlastnosti	Negativní vlastnosti
Funkční i v přepínaných a vysokorychlostních sítích	Delegace IDS až na úroveň jednotlivých elementů
	Složitější integrace a správa

Tabulka 1.5: Výhody a nevýhody NNIDS

1.2.2 Struktura IDS

Systémy detekce průniku lze kategorizovat z hlediska jejich struktury na centralizované a distribuované.

1.2.2.1 Centralizované IDS

Centralizované IDS pracují jako samostatné, nezávislé jednotky, které spolu vzájemně nekomunikují. Data naměřena jedním nebo více IDS jsou často transportována do centrálního prvku, kde je prováděna detailní analýza [7]. Centralizované IDS se díky své jednodušší koncepci vyskytují například ve firewallech nebo jiných síťových hardwarových prostředcích. Centralizované systémy poskytují v mnoha případech detailnější informace o případných zjištěných problémech, ovšem jedná se o zachycené útoky a aktivity lokálního charakteru.

Pozitivní vlastnosti	Negativní vlastnosti
Jednodušší implementace	Detekce útoků lokálního charakteru
Možná integrace přímo v HW sítí	

Tabulka 1.6: Výhody a nevýhody centralizovaných IDS

1.2.2.2 Distribuované IDS

Distribuované IDS se skládají z více IDS, které spolu vzájemně komunikují pomocí sítě. Používají ke své činnosti agentů, kteří jsou lokalizováni na jednotlivých místech sítě. Agenti jsou samostatní a sbírají potřebná data, dle filozofie systému mohou nasbíraná data vyhodnocovat a případně vykonávat náležitá opatření v reakci na zjištěný problém. Každý detekční systém data samostatně vyhodnocuje a sdílí s ostatními spolupracujícími systémy výsledky, které byly zjištěny. Díky sdílení vyhodnocených dat pak získává celý systém IDS celistvější informace o dění v systému a tak distribuované systémy

mohou odhalovat i specifičtější útoky, které probíhají ve více lokalitách systému současně. Pomocí distribuovaných IDS jsou velmi dobře detekovatelné distribuované útoky typu DoS a DDoS [7].

Pozitivní vlastnosti	Negativní vlastnosti
Větší komplexnost než u centralizovaných systémů	Složitější správa
Dobré odhalování distribuovaných útoků	Složitější implementace

Tabulka 1.7: Výhody a nevýhody distribuovaných IDS

1.2.3 Principy detekce IDS

Důležitou vlastností systémů detekce průniků je schopnost odlišit standardní chování uživatelů, aplikačních programů a systému od chování abnormálního, které může zapříčinit narušení bezpečnosti nebo integrity systému a dat. V praxi vzniklo více přístupů k této problematice, které se snaží o co nejpřesnější detekci. Detekce nežádoucího chování je ovšem velmi obtížná, neboť chování jednotlivých uživatelů bývá často nedeterministické a bývá ovlivněno nejrůznějšími vnějšími i vnitřními faktory. Činnosti jsou klasifikovány na základě detekce anomálií, která vyhodnocuje nenadálé změny ve standardním chování subjektu nebo na základě znalosti vzorů, kdy je chování porovnáváno se vzory známých útoků a podezřelých aktivit. Objevují se i další metody, které mohou vyhodnocovat souvislosti mezi vzájemně provázanými jevy a určovat tak míru rizikovosti probíhajících aktivit [8].

V praxi se nejčastěji používají systémy založené na porovnávání vzorů. Tyto systémy jsou méně náročné na administraci, nevyžadují zdlouhavé procedury tvorby modelů chování a eliminují občasné servisní odstavení systému nezbytné pro aktualizaci vzorů chování v systémech založených na detekci anomálií.

Pro větší komplexnost se používají systémy korelační, které analyzují data z různých typů senzorů propojených do jednoho celku a detekují útoky a nebezpečné situace na základě souvislostí mezi probíhajícími ději.

1.2.3.1 Detekce statistických anomálií

Pro metody založené na detekci statistických anomálií jsou základním stavebním kamenem vzory chování. Tyto vzory reflektují běžné chování jednotlivých uživatelů, aplikací i celých provozovaných systémů. Principem je vytvoření vzoru normálního chování pro daný subjekt, se kterým následně bude porovnáváno aktuální chování a vyhodnocovány anomálie vůči [8].

Vzory chování je nutno nejdříve vytvořit, což může být velmi problematické a časově náročné, neboť je potřeba postihnout značné množství běžně vykonávaných aktivit. Je nutno zajistit, aby vzory byly vytvářeny skutečně z normálního chování, což může způsobovat velké problémy ohledně spolehlivosti této metody. Například když uživatel záměrně při získávání vzoru provádí škodlivé činnosti nebo je systém napaden škodlivým softwarem, pak při detekci pomocí IDS systému nebude ani v případě škodlivého chování zjištěna anomálie a akce budou skryty pozornosti správce.

Chování uživatelů je možno charakterizovat pomocí následujících ukazatelů:

- posloupnosti akcí, které uživatelé pravidelně provádějí, např. přihlášení do systému, následuje spuštění účetního software,…
- míra využití systémových prostředků
- objemy přenášených dat po síti
- lokality, ze kterých se uživatelé vzdáleně přihlašují
- časy, kdy jsou uživatelé aktivní
    - čas, kdy se uživatel přihlašuje do systému

    - čas, kdy je uživatel odhlášen
    - průměrné množství času stráveného prací se systémem
- procesy a služby, které uživatelé pravidelně využívají

Chování aplikací lze charakterizovat například podle následujících ukazatelů:

- míra využití systémových prostředků
    - využití paměti
    - využití procesoru
    - počty síťových spojení
    - počty zápisů a čtení na pevný disk
    - velikost dat přenášených po síti
- vlastník procesu
- umístění, ze kterého je aplikace spuštěna
- úroveň práv, se kterými je aplikace spuštěna
- délka běhu aplikace
- čas spuštění aplikace
- analýzou logu zjištěné typické posloupnosti činností aplikace

Chování systému je možno zaznamenat na základě následujících ukazatelů:

- seznamy běžících procesů a služeb
    - počty běžících služeb a procesů
    - seznam procesů a služeb, které jsou obvykle používány
- počty přihlášení k systému
    - počty přihlášených uživatelů
    - počty úspěšných a neúspěšných pokusů o přihlášení k systému
- integrita konfiguračních souborů systému
- přístupy k auditním souborům
- četnost komunikace mezi procesy
    - počty zasílaných zpráv aplikacemi
- míra využití systémových prostředků
    - využití paměti
    - využití procesoru
    - počty síťových spojení
    - počty zápisů a čtení na pevný disk
    - velikost dat přenášených po síti

Pomocí zaznamenávání a analýzy těchto ukazatelů je možno vytvořit vzory chování a posléze je během fáze monitorování porovnávat s aktuálním chováním a vyhodnocovat tak odchylky vůči vzorům. Vzory chování je nutno postupně adaptovat nebo vytvořit nové vzory, protože chování jednotlivých subjektů se čas od času mění v závislosti na jejich potřebách.

Ke své činnosti systémy detekce anomálií využívají celou řadu metod, kterými anomálie detekují. Těmto metodám se věnuje čtvrtá kapitola tohoto textu a zde budou jen velmi stručně zmíněny některé z nich [4]:

- Práh detekce se používá k odlišení normálního a abnormálního chování v případě sledování vlastností systému nebo systémových zdrojů, které jsou reprezentovány pomocí číselných údajů. Může být určena míra, po kterou je tato hodnota klasifikována jako normální. Technika určení prahu se používá například při sledování využití paměti, procesoru nebo počtu přihlášení do systému.
- Statistické měření ať už parametrické, kdy rozložení profilovaných atributů odpovídá vzorům nebo neparametrické, kdy se rozložení profilových atributů učí z již dříve naměřených hodnot.
- Měření založené na pravidlech, které je podobné neparametrickému statistickému měření. Rozhoduje se pomocí aplikace těchto pravidel na naměřená data o tom, zda jsou v mezích normy nebo značí abnormální chování. Na rozdíl od předchozí metody se nezabývá číselnými hodnotami, ale daty ve formě pravidel.
- Rozvíjí se používáni genetických algoritmů, neuronových sítí nebo modelů imunitního systému.

Silnou stránkou systémů založených na detekci statistických anomálií je jejich unikátní schopnost odhalovat i útoky a hrozby, které v minulosti nebyly nikdy zaznamenány a popsány. Na druhou stranu ovšem kvůli komplikovanosti procesu detekce anomálií a typické vlastnosti, že vše, co neodpovídá vzoru chování, je považováno za podezřelou činnost, hrozí, že při přílišné citlivosti velké množství falešně pozitivních varování může způsobovat i ignoraci útoků, které skutečně probíhají a mají potenciál uškodit [2].

Pozitivní vlastnosti	Negativní vlastnosti
Schopnost detekce nových a neznámých útoků	Značné množství falešných poplachů vyvolaných neškodným, ale nedeterministickým chováním uživatelů nebo systému
Menší závislost IDS na vnějším prostředí	Nutné dlouhé časové období k vytvoření profilu normálního chování uživatele nebo systému
Není nutné detailně znát důvody a vnitřní strukturu chování uloženého ve vzorech	Chování uživatelů se může měnit v čase a je nutno systém těmto změnám pravidelně přizpůsobovat
Schopnost detekce narušení úrovně uživatelských práv	Je nezbytné zamezit nebezpečnému chování subjektů ve fázi vytváření vzoru a učení systému

Tabulka 1.8: Výhody a nevýhody metody detekce statistických anomálií [2]

1.2.3.2 Porovnávání vzorů

Systémy využívající porovnávání vzorů bývají velmi často provozovány jako systémy detekce průniků v reálném čase, neboť na rozdíl od metody detekce anomálií mají přesně definované vzory, se kterými aktuální chování systému, aplikací nebo uživatelů porovnávají. Systému jsou k dispozici údaje o útocích a nebezpečném chování ve formě vzorů, na základě kterých může IDS tyto útoky detekovat [8]. Na rozdíl od detekce anomálií jsou všechny aktivity, které neodpovídají vzorům, hodnoceny systémem jako neškodné, což zapříčiňuje velmi malou možnost zisku falešně pozitivních poplachů. Na druhou stranu není IDS schopen zachytit nové útoky a rozpoznat neznámé hrozby od korektně probíhajících aktivit systému či uživatelů.

Vzory jsou často založeny na principech konečných automatů, rozhodovacích stromů, expertních systémů, neuronových sítí a pravděpodobnostních modelů.

Těmito IDS systémy je možno detekovat nesrovnalosti v případě, že jsou známé vzory nebezpečného chování zejména v těchto oblastech:

- Podezřelé záznamy v systémových konfiguračních souborech
- Nebezpečné běžící procesy a služby
- Nebezpečné položky v registru operačního systému
- Podezřelé záznamy v tabulce síťových spojení
- Nebezpečné sekvence příkazů zadávané uživateli

Vzory je možno obecně rozdělit do dvou základních kategorií:

- Vzorky textových řetězců: textové řetězce detekované v datech mohou provázet netypické chování. Například volání příkazu "sudo", kdy je detekováno použití vysoké úrovně uživatelských práv.
- Popis útoků: popis časového průběhu na sebe navazujících činností, které mohou při konkrétním uspořádání vykonávat nebezpečnou aktivitu.

Pozitivní vlastnosti	Negativní vlastnosti
Malé množství falešných poplachů	Není možno detekovat neznámé útoky
Existence vzorů pro většinu známých útoků	Možnost maskování škodlivé aktivity oklamáním senzorů
Snadné vytváření vzorů a jednoduché algoritmy detekce	Nutná pravidelná údržba IDS a databáze vzorů
Nenáročné z hlediska systémových prostředků	
Vhodné pro sledování v reálném čase	

Tabulka 1.9: Výhody a nevýhody metody porovnávání vzorů [2]

1.2.3.3 Korelační systémy

Systémy založené na korelaci vyhledávají souvislosti mezi jevy probíhajícími na více místech. Korelační IDS bývají většinou tvořeny sítí agentů a centrálního prvku, který vyhodnocuje nastalou situaci. IDS tedy agreguje data z více zdrojů a na základě korelace pak vyhodnocuje nebezpečné situace a případné útoky, o kterých informuje své partnerské systémy [10]. Nejjednodušším příkladem může být souvislost aplikačního vybavení a operačního systému na kterém běží. Je možno například rozlišit nebezpečnou situaci při používání aplikace na platformě Microsoft Windows, ale naopak stejná situace při požití stejné aplikace na platformě Unix nezpůsobuje bezpečnostní problém díky jiné architektuře systému. Jedním z největších problémů metody je korelace v čase, kdy je potřeba kromě souvislostí mezi událostmi brát v úvahu také jejich časovou dimenzi. Vzhledem ke své složitosti jsou systémy založené na této bázi spíše ve fázi výzkumu. K analýze získaných dat se často používá metod model-checkingu.

Pozitivní vlastnosti	Negativní vlastnosti
Malé množství falešných poplachů	Složité nasazení
Účinná detekce útoků a hrozeb	

Tabulka 1.10: Výhody a nevýhody korelační metody

1.2.4 Zdroje dat pro IDS

Pro systémy detekce průniků je klíčovým faktorem dostatek validních informací. Tyto informace jsou výsledkem analýzy shromážděných dat, která mohou pocházet z různých zdrojů a mohou být získávána rozličnými způsoby.

1.2.4.1 Detekce na základě analýzy stavu systému

Systémy IDS, které pracují na bázi analýzy stavu, ve kterém se systém nachází, jsou zaměřeny na monitoring způsobu využití systémových prostředků a na kontrolu integrity důležitých systémových souborů a registrů systému v případě, že je systém implementuje. Činnost IDS je zaměřena na pořizování snímků systému v čase a jejich porovnávání s aktuálním stavem. Při detekci změn dochází k jejich analýze a při klasifikaci těchto změn jako nebezpečných nebo riskantních je iniciována akce, kterou může být upozornění správce systému v případě pasivních systémů nebo vykonání akcí vedoucích k nápravě v případě aktivních IDS. Změny v systémových a konfiguračních souborech mohou zapříčiňovat jak legální akce uživatelů, kteří manipulují se systémem běžným způsobem a instalují například nové aplikace, tak i škodlivý software nebo útočníci s motivem poškodit systém, deaktivovat ochranné mechanismy systému nebo ukrýt se. Proto je důležitá analýza těchto změn a stanovení rizika jimi způsobeného.

Dalším aspektem, který může být sledován, je způsob využití systémových prostředků, jako je alokace paměti, využití místa datových uložišť nebo zatížení procesoru. Důležité informace mohou poskytnout i seznamy běžících služeb a procesů, ve kterých může IDS detekovat známé škodlivé nebo nebezpečné aplikace nebo aplikace či uživatele způsobující neobvyklé zatížení systémových prostředků.

Pozitivní vlastnosti	Negativní vlastnosti
Vhodné pro sledování v reálném čase	Nedetekuje útoky, které nemění sledované oblasti systému
Jednoduchá implementace	Možnost maskování škodlivé aktivity oklamáním senzorů
Nenáročné z hlediska systémových prostředků	

Tabulka 1.11: Výhody a nevýhody detekce metodou analýzy stavu systému

1.2.4.2 Detekce na základě analýzy síťových paketů

Metody založené na základě analýzy síťových paketů používají systémy detekce průniků NIDS a NNIDS. V této oblasti je uplatňována celá řada různých strategií analýzy. Existující přístupy se často zabývají identifikací útoků a nebezpečných situací na základě detekce pomocí vzorů a systémů, jež analyzují síťový provoz na základě znalosti protokolů používaných v síti.

První přístup umožňuje odhalovat základní útoky, které mohou být zpravidla detekovány například podle konfigurace odchozích a příchozích portů, která se vymyká standardu atypickými rozsahy zdrojových adres nebo díky nesrovnalostem v číslování sekvencí paketů. Jednou ze základních metod identifikace útoků v této skupině metod je detekce nebezpečných paketů, které mohou být odhaleny na základě jejich flagu nebo atypické velikosti, kdy je detekčním systémem možno například odlišit škodlivé Ping of Death od neškodných paketů [2].

Druhý přístup analyzuje pakety na vyšších vrstvách, díky čemuž umožňuje detekovat specifičtější a sofistikovanější útoky. V těchto vyšších vrstvách síťové architektury je možno analyzovat pakety na přítomnost prvků typických pro sledované komunikační protokoly, ovšem za cenu vyššího využití systémových prostředků při detekci a nutnosti implementovat propracovanější algoritmy detekce, které musejí být schopny analyzovat konkrétní protokoly a musejí přesně znát jejich vlastnosti. Touto metodou je například možno detekovat DoS útoky směřované na POP3 protokol, kdy je sledována četnost požadavků k vykonání příkazů, které mohou zahltit server.

Detekce na základě analýzy síťových paketů má slabiny v případě vysoké rychlosti přenosu dat sítí, kdy IDS nemusí stíhat pakety analyzovat a některé útoky mohou zůstat nedetekovány. Dalším zdrojem problémů je šifrovaná komunikace, která nemůže být analyzována a dále pakety, které jsou fragmentovány, takže systém je nemusí být schopen analyzovat jako jeden nefragmentovaný celek.

Pozitivní vlastnosti	Negativní vlastnosti
Vhodné pro sledování v reálném čase	Pro detekci na vyšších vrstvách nutnost znalosti komunikačních protokolů
Účinné proti známým útokům a hrozbám	Problematické při fragmentaci paketů a šifrované komunikaci
	Problematické při vysokorychlostní komunikaci

Tabulka 1.12: Výhody a nevýhody detekce metodou analýzy paketů

1.2.4.3 Detekce na základě analýzy auditních dat

Mezi nejvíce využívané metody získávání dat k detekci útoků patří analýza auditních dat, která jsou shromažďována operačními systémy ve formě logovacích souborů. Tyto soubory záznamů koncentrují údaje o aktivitách systému, aplikací a uživatelů, které byly vykonány. Záznamy jsou chronologicky seřazeny. Soubory auditních dat bývají velmi rozsáhlé, což značně komplikuje jejich zpracování lidmi a proto je zde namístě využití IDS systémů, jejichž cílem je automatická detekce nežádoucích a podezřelých aktivit. V některých případech je analýza auditních dat jedinou cestou, jak detekovat uskutečněný útok, protože nemusí být přítomny žádné další viditelné příznaky o jeho uskutečnění. Touto technikou je možno detekovat i neobvyklé chování systému, které nemusí být nutně zapříčiněno útoky, ale může se jednat například o neoprávněné využívání výpočetního prostředku nebo o pokusy překročit přidělená uživatelská oprávnění. Auditní data bývají archivována a tak je možno příčiny problémů zjistit i zpětně [11].

Problémem u auditních dat je určení doby, po kterou je vhodné je uchovávat k možnosti zpětné analýzy, neboť bývají značně rozsáhlá a zabírají prostor na datových uložištích. Auditní data je nutno chránit před modifikací a prohlížením neoprávněnými osobami, protože by se útočník nebo škodlivá aplikace mohly pokoušet maskovat stopy po útoku nebo škodlivé činnosti úpravou souborů obsahujících auditní data. V případě možnosti čtení auditních dat neoprávněnými osobami hrozí, že potencionální útočník může skrytě získávat informace o struktuře a fungování systému a využít získané znalosti později k útoku.

Pozitivní vlastnosti	Negativní vlastnosti
Auditní data vytváří systém	Velký objem auditních dat
Umožňuje detekci v dávkách	Nutná ochrana auditních dat před modifikací a neautorizovaným prohlížením
Detekce útoků a nebezpečných činností i zpětně	Složitá identifikace podstatných dat

Tabulka 1.13: Výhody a nevýhody detekce metodou analýzy auditních dat [11]

1.2.5 Chování IDS při detekovaném útoku

Z pohledu reakce systémů detekce průniků při zjištění problémů je možno definovat dvě kategorie, do kterých systémy spadají. Na jedné straně jsou běžné systémy, které zastávají pouze funkci monitorování útoků a shromažďování informací o nich, na druhé straně pak systémy schopné automatické reakce na útok.

1.2.5.1 Aktivní IDS

Systémy, které jsou navrženy a implementovány jako aktivní, jsou schopny kromě detekce útoků a hrozeb také automaticky reagovat. Cílem reakce je ochrana sledovaného systému, zamezení získání dat útočníkem nebo rekonfigurace systému tak, aby útok vedený v budoucnu stejným postupem byl neúspěšný [11]. Pro aktivní chování je nutné, aby systém znal správné reakce na konkrétní nastalou situaci, což nemusí být vždy jednoduché a zcela jednoznačné. Aktivní systémy musí být vybaveny z tohoto důvodu bází znalostí a složitými algoritmy rozhodování, které vyhodnocují nastalou situaci.

Pozitivní vlastnosti	Negativní vlastnosti
Bezprostřední reakce na útok	Složité systémy
Inteligentnější a sofistikovanější než pasivní systémy	Možnost špatné reakce na událost
Účinné, pokud znají dobře prostředí a mají velkou databázi znalostí	

Tabulka 1.14: Výhody a nevýhody aktivních IDS

1.2.5.2 Pasivní IDS

Většina současných IDS se řadí do skupiny pasivních systémů. Jejich implementace a používané algoritmy jsou jednodušší, než v případě systémů aktivních. Jejich účelem je pouze zaznamenat nežádoucí aktivity a podat o nich náležitou formou zprávu správci, který po analýze dat o útoku vykoná nápravu, identifikuje škody vzniklé útokem a pokud je to reálné, tak vhodnými prostředky zamezí podobným útokům v budoucnosti. Ke komunikaci se správcem systémy využívají různých forem, mezi které se řadí SNMP, email nebo automatizované zasílání SMS zpráv na mobilní telefon [11].

Pozitivní vlastnosti	Negativní vlastnosti
Jednodušší a méně náročné než aktivní systémy	Nutnost fyzického zásahu správce při reakci na útok
Jednoduší implementace	Nápravná opatření je možno provádět až se zpožděním

Tabulka 1.15: Výhody a nevýhody pasivních IDS

1.2.6 Rozdělení IDS podle okamžiku vyhodnocování

Systémy detekce průniku jsou obecně založeny na dvou přístupech z hlediska časového okamžiku analýzy shromážděných dat. Rozdělení do těchto kategorií je zapříčiněno jednak volbou konkrétní metody detekce dat a jednak nároky kladenými správci detekčního systému.

1.2.6.1 Vyhodnocení v reálném čase

Systémy provádějící analýzu v reálném čase jsou z pravidla určeny pro online sledování důležitých systémů a aktivit uživatelů. Typickým nasazením je detekce útoků v rámci síťové komunikace. Zde bývá permanentně sledován veškerý provoz, který prochází IDS systémem. K detekci se často využívá vzorů známých útoků a nebezpečných aktivit, pomocí kterých je možno identifikovat nebezpečné pakety nebo podezřelé konfigurace otevíraných síťových portů, které nejsou standardní. Často se používá také detekce známých nebezpečných textových řetězců. Omezení vyhodnocování

v reálném čase závisí na výkonnosti detekčního systému, kdy při přetížení IDS dochází k ignorování některých naměřených dat a je tak možné, aby některé známé útoky pronikly bez varování. Oproti dávkovým systémům je využíván jen velmi malý nebo žádný prostor k uložení dat, ale naopak je požadováno velké množství paměti pro aktuálně zpracovávaná data. Pro vyhodnocování v reálném čase je charakteristické použití rychlých a jednoduchých algoritmů. Díky okamžité detekci útoku je možno nejen útok zaznamenat, ale i adekvátně na něj zareagovat a znemožnit jej nebo po útoku uskutečnit automatické kroky vedoucí k nápravě nebo zotavení systému v případě aktivních IDS [2].

Pozitivní vlastnosti	Negativní vlastnosti
Pokrývají velkou oblast	Analyzují jen část informací, které jsou aktuálně dostupné
Bezprostřední reakce a rychlé zotavení po útoku	Vyšší náročnost na výpočetní výkon při velkém množství zpracovávaných dat
Možnost obrany proti DoS útokům	
Malé nároky na úložiště analyzovaných dat	

Tabulka 1.16: Výhody a nevýhody vyhodnocování dat v reálném čase

1.2.6.2 Dávkové vyhodnocování

Systémy s dávkovým vyhodnocováním shromážděných dat jsou převládajícím typem v oblasti běžně používaných IDS. Dávkové zpracování definuje potřebu shromažďovat data k pozdější analýze, což s sebou nese celou řadu komplikací, se kterými se IDS musí vyrovnat. Prvotním problémem je rozhodnutí o granularitě shromažďovaných dat, která je zásadní jak pro schopnosti detekce, tak pro míru zatížení sítě a monitorovaného systému. Při velkém množství uchovávaných dat hrozí zahlcení komunikační sítě, případně neúměrné zatížení výpočetních prostředků při pokusech o zmenšení velikosti shromažďovaných dat užitím komprese. Kritickým faktorem je výběr správného typu shromažďovaných dat, neboť není vždy možno jednoznačně určit, která data jsou podstatná a budou v budoucnu důležitá pro analýzu. Dalším důležitým faktorem je doba uchování auditních dat. Z hlediska bezpečnosti je nutno auditní data dobře chránit, aby nemohlo docházet k jejich neautorizované analýze potencionálními útočníky, kteří by mohli získat některé citlivé informace a získaných znalostí využít při útoku na systém [3].

Záznam dat k analýze musí splňovat jisté požadavky. Jedná se zejména o minimalizaci zatížení systémů způsobeného sběrem a uchováním dat, odolnost proti DoS útokům, které jej mohou ohrozit, využití mechanismů zpracování dat jako agregace, umělá inteligence nebo dolování dat.
Dávkové zpracování se využívá zpravidla kvůli [3]:

- Detekci opakujících se nežádoucích aktivit
- Identifikaci úspěšných útočníků
- Identifikaci nedostatků systému
- Vytváření modelů chování uživatelů, aplikací a systémů pro účely systémů s detekcí anomálií
- Možností identifikace útoku zpětně
- Ochraně legálních uživatelů systému

Pozitivní vlastnosti	Negativní vlastnosti
Nízká zátěž systému	Velký objem ukládaných dat
Možná manuální analýza dat	Obtížný výběr dat k archivaci
Snadněji odhalitelné útoky, které se opakují	Nutnost chránit shromážděná data proti zneužití
	Není možná bezprostřední reakce

Tabulka 1.17: Výhody a nevýhody vyhodnocování dat po dávkách

1.3 Zhodnocení IDS systémů

Současné trendy v oblasti bezpečnosti výpočetních systémů naznačují směr, kterým by se mohl vývoj IDS ubírat. Jako perspektivní se v praxi jeví ze systémů detekce průniků učinit jeden nebo několik bezpečnostních prvků v komplexním bezpečnostním systému. Ve spolupráci se stávajícími síťovými firewally, filtrujícími směrovači, antivirovými a antimalware systémy mohou dosáhnout značné efektivity a pomohou ochránit systémy a sítě proti útokům, vůči kterým by byly současné bezpečnostní mechanismy bezmocné.

Pro správnou funkčnost IDS systémů je kritická zejména jejich správa a korektní nastavení, kdy hrozí při chybách konfigurace velké množství falešně pozitivních hlášení o útocích nebo naopak propouštění útoků a ponechání nebezpečných aktivit bez povšimnutí.

Pozitivní vlastnosti	Negativní vlastnosti
Poskytují hloubkovou ochranu	Množství falešně pozitivních poplachů
Účinný doplněk stávajících bezpečnostních systémů	Nákladné a komplikované
Centrální správa	Nutnost dohledu zkušených administrátorů
Možnost automatické reakce	Nutnost analyzovat velké množství dat
Detailní informace o slabinách systému a provedených útocích	Na útok většinou reagují zpětně
Detekce interních i externích útoků	Většina systémů je náročná na využití systémových prostředků

Tabulka 1.18: Výhody a nevýhody IDS

2 Seznámení s existujícími HIDS

V kapitole budou přiblížena některá z existujících řešení, budou popsány jejich základní vlastnosti a oblasti, na které soustřeďují svoji činnost.

2.1 IDS OSSEC

OSSEC je bezpečnostní, volně šiřitelný HIDS systém pod licencí GNU. Z hlediska platformy se jedná o multiplatformí řešení určené pro MS Windows, Apple OS X, BSD a systémy založené na jádře Linuxu.

OSSEC se zaměřuje na detekci průniků pomocí analýzy systémových souborů, kontrolu integrity souborů a důležitých systémových oblastí, monitoring systémových politik a detekci rootkitů. OSSEC pracuje v reálném čase a umožňuje provádět na základě zjištěných útoků a abnormálních činností aktivní kroky k nápravě nastalé situace.

OSSEC se vyznačuje zejména velmi silnou analýzu logů, když v této oblasti podporuje kromě auditních záznamů vytvářených operačními systémy i celou řadu externích aplikací, které vytvářejí záznamy o svém provozu. Jedná se zejména o rozšířenější webové, poštovní, databázové a ftp servery, firewally a bezpečnostní nástroje. OSSEC umožňuje i spolupráci s NIDS systémy společností Cisco a IDS Snort, což dovoluje vytvořit komplexnější a účinnější IDS systém.

Z pohledu architektury se jedná o centralizované řešení, kdy na klientských stanicích běží agenti sbírající data, která jsou transportována na centrální konzolu. Centrální konzola je realizována aplikací běžící na pozadí operačního systému, ke které je možno přistupovat pomocí příkazů textové příkazové řádky nebo pomocí volitelného webového rozhraní, které je možno v případě zájmu doinstalovat.

Obrázek 2.1: Uživatelské prostředí HIDS OSSEC

2.2 IDS Samhain

Samhain je volně šiřitelným HIDS systémem. Tento IDS systém je multiplatformní a je určen pro instalaci na celou řadu operačních systémů.

HIDS Samhain může být použit jak pro samostatné počítače, tak i v síti více počítačů, kde nabízí centrální správu a monitoring. Systém umožňuje správcem definovanou aktivní odezvu v případě zaznamenání útoku.

Tento IDS se zaměřuje převážně na monitoring integrity operačního systému a systémových souborů. Při kontrole integrity souborů jsou využívány techniky různé úrovně od nejjednodušších, realizovaných vytvářením a kontrolou kontrolních součtů souborů, přes analýzu atributů až po analýzu skrytých souborů kvůli hrozbě ukrývání rootkitů. Při sledování integrity systému je sledována celá řada systémových oblastí, jsou analyzovány soubory logů systému, kontrolována integrita jádra, monitorovány pokusy o přihlášení k systému, sledovány otevřené síťové porty a mnohé další vlastnosti [23].

Samhaim umožňuje integraci a spolupráci s ostatními IDS systémy, je vhodný jako integrovatelná součást systému Prelude.

Ovládání systému je možno prostřednictvím textové konzoly, případně může být pro vyšší komfort grafického uživatelského rozhraní integrován do systému Prelude [23].

2.3 IDS TripWire

TripWire je v současnosti volně šiřitelným produktem společnosti TripWire, která nabízí mimo jiné i jeho komerční variantu, poskytující vyšší funkcionalitu, zaměřenou především na nasazení ve velkých podnikových sítích.

IDS TripWire je zaměřen na systémy na bázi Unixu a Linuxu a spadá do kategorie HIDS, instalovaných na samostatných stanicích se zaměřením na oblast kontroly integrity souborového systému [24]. U jednotlivých monitorovaných souborů jsou sledovány následující parametry:

- přístupová práva
- číslo i-uzlu
- počet odkazů na soubor
- UID (ID uživatele - vlastníka souboru)
- GID (ID skupiny - vlastníka souboru)
- typ souboru
- velikost souboru
- počet alokovaných datových bloků souboru
- čas modifikace souboru
- čas modifikace i-uzlu
- čas přístupu

TripWire běží na pozadí operačního systému a jeho konfiguraci a administraci je možno provádět pomocí příkazové řádky.

3 Detekce útoků HIDS

Tato kapitola přiblíží běžné typy útoků a mechanismy systémů HIDS, které je umožňují detekovat. Budou zde zmíněny základní vlastnosti a informace, které je nutno analyzovat pro úspěšnou detekci útoků a nebezpečných aktivit.

3.1 Princip detekce útoků

V oblasti systémů IDS existuje celá řada přístupů, jak útoky detekovat. Z hlediska principu zpracování lze obecně odlišit dvě kategorie metod, které již byly detailně popsány v první kapitole této práce. Jedná se o metody založené na detekci statistických anomálií dat a o metody porovnávání vzorů s aktuálně probíhajícími činnostmi. S ohledem na tento fakt je nutno přizpůsobit vlastnosti sledované systémem a přizpůsobit doménu parametrů, ze které jsou detekovány události a informace využívané IDS ke své činnosti.

3.2 Důležité zaznamenávané vlastnosti pro detekci útoků v OS Microsoft Windows

S ohledem na výše zmíněné vlastnosti detekčních systémů, popsané v první části této kapitoly, byl vytvořen seznam parametrů, které je nutno zaznamenávat pro úspěšnou detekci útoků a nebezpečných aktivit. Protože cílem praktické části, navazující na tyto kapitoly je návrh a implementace HIDS systému pro OS Microsoft Windows, budeme se zde zabývat pouze parametry, validními v tomto prostředí.

3.2.1 Ukazatele zatížení systémových prostředků

Sledování ukazatelů zatížení systémových prostředků je zejména vhodné pro systémy detekující anomálie dat. Díky nim lze rozpoznat změny v typickém chování, které mohou implikovat napadení systému škodlivým softwarem, případně nežádoucí aktivity uživatelů. Jako vhodní se jeví ukazatelé zatížení procesoru či pevného disku. Jako méně vhodní pak ukazatelé zatížení paměti, neboť moderní operační systémy implementují různé strategie jejího přidělování. K získání detailních informací o aktuálním zatížení systémových prostředků je možno použít rozhraní Windows API a službu WMI.

3.2.2 Integrita systémových souborů a důležitých oblastí systému

Integrita systémových souborů a oblastí je důležitým ukazatelem pro monitoring útoků. Je možno sledovat přístupy do významných adresářů a souborů a například pro kriticky důležité soubory vytvářet kontrolní součty a tak sledovat jejich modifikaci. Tato metoda je vhodná k zjištění škodlivých aplikací a uživatelů provádějících škodlivou činnost.

3.2.3 Seznamy běžících procesů a služeb

Pro účely HIDS systémů mohou být zaznamenávány seznamy procesů a služeb běžících v rámci operačního systému. Je možno sledovat počty procesů a služeb jako základní parametr a jako rozšíření monitorovat jejich konkrétní názvy, původ a míru, jakou využívají systémové prostředky.

Na základě běžících procesů a služeb je možno pomocí systémů detekce anomálií detekovat změny v chování uživatelů nebo aplikací, pomocí systémů porovnávajících vzory lze určit škodlivé aplikace a služby systému. K informacím o procesech a službách vede v operačních systémech Windows cesta využitím služeb Windows API a WMI, které jsou detailně popsány v následující kapitole.

3.2.4 Aktivita síťové komunikace

Důležitým parametrem s vysokou vypovídací hodnotou je aktivita komunikace pomocí sítě a tabulka navázaných síťových spojení. Na jejím základě je možno v případě systémů detekce anomálií detekovat například trojské koně, škodlivý software rozesílající spam nebo nelegální počínání uživatelů. V případě systémů porovnávajících vzory je možno naleznout nesmyslné kombinace zdrojových a cílových portů, jež indikují podezřelou činnost. K podrobným údajům o síťových spojeních je možno přistupovat poměrně snadno pomocí služby systému WMI.

3.2.5 Přístupy ke sdíleným prostředkům

Přístupy ke sdíleným prostředkům po síti můžou indikovat problémy s narušením bezpečnosti v rámci sítě, kdy se uživatelé pokoušejí neoprávněně získat sdílená data. Informace o sdílených prostředcích lze nejsnadněji získat pomocí služby WMI, která poskytuje značné množství detailů v této oblasti nebo analýzou logů, kde se jednotlivé pokusy o přístup zaznamenávají.

3.2.6 Data získaná analýzou auditních a konfiguračních souborů

Z hlediska systémů detekujících anomálie dat je možno ze souborů obsahujících auditní data získat model chování dané aplikace nebo systému a v budoucnu jej porovnávat s aktuálním stavem. U systémů IDS detekujících pomocí vzorů je možno v auditních datech a konfiguračních souborech nalézt nebezpečné záznamy, které implikují nebezpečnou situaci nebo uskutečněný útok. Data se získávají případně i z registru operačního systému.

3.2.7 Komunikace mezi procesy

Podle množství komunikace mezi procesy, která se v operačním systému Windows uskutečňuje zasíláním zpráv, je možno v systémech detekce anomálií vytvořit model běžného chování aplikací a systému a na základě odchylek detekovat abnormální chování, které může naznačovat virovou infekci nebo zamoření systému škodlivými aplikacemi. Údaje o komunikaci mezi procesy lze získat prostřednictvím rozhraní Windows API nebo WMI.

3.2.8 Přihlašování a odhlašování od systému

Četnosti přihlašování a odhlašování od systému a počty přihlášených uživatelů mohou být užitečné k vytvoření modelu chování v systémech detekce anomálií, kdy mohou být zaznamenány například pokusy o prolomení hesla nebo neautorizované pokusy o přihlášení do systému. Informace o přihlašování do systému a přihlášených uživatelích je možno získat pomocí Windows API, WMI a analýzou auditních dat.

3.2.9 Akce prováděné s vysokým uživatelským oprávněním

Monitorování akcí, ke kterým je nutno použít vysoké oprávnění, je užitečné jak v systémech detekce anomálií, kdy je možno rozlišit abnormální chování uživatele od standardního, tak i v případě

porovnávání vzorů, kdy může být detekována nebezpečná sekvence příkazů ohrožující bezpečnost. Informace o akcích prováděných s vysokým oprávněním je možno získat pomocí služby WMI nebo analýzou auditních dat.

3.2.10 Časová období aktivity

Ukazatelem s dobrou vypovídací hodnotou mohou být časové údaje o aktivitách uživatelů nebo aplikací, kdy je možno sledovat doby běhu systému a aplikací nebo časy přihlášení uživatelů. Z těchto dat je možno vytvořit model typického chování a na základě detekce anomálií pak vyhodnocovat, zda jsou aktivity korektní nebo zda jsou podezřelé.

4 Prostředky systému MS Windows pro získání dat k detekci útoků

Kapitola se zabývá způsoby a prostředky, umožňujícími získávání nezbytných informací v prostředí operačních systémů Microsoft Windows, na jejichž základě mohou být detekovány útoky a atypické chování systému, aplikací a uživatelů.(kapitola převzata z [16])

4.1 Windows API

4.1.1 Charakteristika Windows API

Windows API představuje systémové rozhraní, vyvinuté společností Microsoft a integrované jako jedna z nedílných součástí do operačních systémů Windows. Účelem Windows API je umožnit co nejjednodušší a nejrychlejší cestou přístup k systémovým funkcím, službám a proměnným systému, jež nabízejí cenné informace. Důležitým posláním tohoto systémového rozhraní je možnost částečného odstínění systému od aplikací, které jsou díky tomu kompatibilnější a nejsou tolik závislé na používané verzi systému. Díky standardizaci Windows API je vývoj aplikací méně finančně náročný a je podstatně jednodušší a rychlejší. Všechny aplikace s výjimkou konzolových aplikací musí s operačním systémem komunikovat právě přes Windows API. Toto rozhraní bylo navrženo pro využití v programovacích jazycích C a C++, ale není problém s ním pracovat i pomocí jiných programovacích jazyků ať již Windows API nativně podporují nebo za využití podpůrných knihoven.

4.1.2 Příklad monitorování vlastností systému pomocí Windows API

Windows API nabízí mnoho nástrojů, pomocí kterých lze získávat informace o monitorovaném systému. Jsou implementovány funkce, s jejichž pomocí mohou aplikace standardizovaným způsobem získávat informace o hardwarových součástech počítače a mohou přistupovat k procesům běžícím v systému a ovládat je. Z funkcí užitečných pro monitorování systému lze jmenovat funkce pro získání zatížení paměti, procesoru a obsazení místa na pevném disku. Mezi hlavní výhody těchto funkcí patří jejich rychlost, díky které jsou ideální pro monitorování v reálném čase.

4.1.2.1 Windows API funkce pro práci s procesy

V operačním systému Microsoft Windows lze přistupovat k běžícím procesům za pomoci Windows API. Jednou ze základních funkcí je funkce nazvaná “OpenProcess()”, která vrací popisovač požadovaného procesu za předpokladu, že má aplikace dostatečná oprávnění k němu přistupovat. Po získání ukazatele na proces je možno dostat různorodé informace o zatížení systémových prostředků nebo nad tímto procesem provádět administrativní úkony.

Pro získání informací o využité paměti se využívá funkce “GetProcessMemoryInfo()“, která tyto informace zapisuje do datové struktury nazvané “PROCESS_MEMORY_COUNTERS”. Informace lze získat v případě, že máme dostatečné oprávnění přístupu k procesu. Uživateli typu administrátor je umožněn přístup ke všem procesům, standardním uživatelům jsou zpřístupněny pouze informace o procesech, jejichž jsou vlastníky. Výsledkem funkce je navrácená datová

struktura, naplněná informacemi o aktuálním a maximálním využití fyzické paměti a stránkovacího souboru.

Získání informací o využití procesoru procesem je složitější, než shromažďování informací o paměti. Je nutno opakovaně vzorkovat čas procesoru spotřebovaný procesem a následně pak na základě takto nashromážděných údajů vypočítat využití procesoru dle vztahu 4.1.

$$\frac{(\text{aktualni\_casCPU} - \text{predchozi\_casCPU})}{\text{uplynuly\_}cas} \qquad (4.1)$$

K sestavení kompletního seznamu všech běžících procesů lze využít funkce "CreateToolHelp32SnapShot", která vytváří obraz o aktuálním stavu procesů běžících v systému. Následně lze vyčíslit seznam procesů za použití funkcí "Process32First" a "Process32Next" a s ním získat i základní informace o procesech ve struktuře nazvané "PROCESSENTRY32". K dispozici jsou touto cestou informace o jménu procesu, počtu jeho vláken, identifikačním čísle, identifikačním čísle rodičovského procesu a o prioritě běhu procesu v systému.

Pomocí Windows API je možno za předpokladu, že disponujeme dostatečnými právy, provádět nad procesy operace, jako jsou ukončení procesu nebo změna jeho priority.

4.2 Služba WMI

4.2.1 Základní informace o službě WMI

Operační systémy společnosti Microsoft obsahují již od uvedení Windows NT 4.0 (SP4) jako standardní součást službu s názvem WMI. Do nižších verzí operačních systémů je možno tuto službu dodatečně nainstalovat. WMI je zkratkou pro Windows Management Instrumentation, což je služba určená pro snazší administraci operačního systému. Účelem zavedení služby WMI bylo vytvoření prostředí, které by nebylo závislé na použité verzi operačního systému a umožnilo by jednotným způsobem aplikacím přistupovat k systémovým, jinak nedostupným informacím, automatizovat administrativní úkony a celkově zjednodušit tvorbu aplikací pro správu systému. Velmi často se této služby využívá ve skriptech nástroje PowerShell. Službou WMI je možno spravovat lokální i vzdálené počítačové systémy. Služba využívá CIM objektového databázového modelu, jehož implementace je označována jako "CIM Repository". Jednotlivé části operačního systému, k nimž můžeme pomocí WMI přistupovat, jsou členěny jako jednotlivé COM objekty v databázi. Databáze je rozšiřitelná a softwarové produkty, jako jsou například SQL Server, Exchange Server, Microsoft Office, do ní přidávají vlastní COM objekty, nad nimiž je následně možno provádět administrativní úkony. Počet objektů, poskytujících WMI informace se, s každou další verzí služby zvyšuje. Mezi důležité objekty, zprostředkující systémové informace, lze bezesporu zařadit například tyto [18]:

Objekt	Popis objektu
Win32 Provider	Zabezpečuje přístup a čerstvé informace o stavu systému, jeho nastaveních a proměnném systémovém prostředí.
Performance Counter Provider	Pomocí tohoto objektu lze přistupovat k statistickým systémovým informacím, které se týkají využití a zatížení systémových zdrojů.
Windows Installer Provider	Zprostředkovává přístup ke službám komponenty "Windows Installer" a umožňuje získávání informací o nainstalovaných aplikacích.
Session Provider	Objekt umožňující získání informací o síťové komunikaci a síťových prostředcích systému.

Tabulka 4.1: Objekty zprostředkující informace pro WMI [17]

4.2.2 Vlastnosti služby WMI

Mezi hlavní rysy rozhraní, vytvářeného službou WMI, patří možnost využití jeho vlastností ke snadné automatizaci. To z něj činní silný nástroj při tvorbě skriptů, kdy je vývojář odstíněn od problematiky nízkoúrovňových funkcí systému, ve kterých může jednotným způsobem přistupovat k systémovým informacím a funkcím. Na informace, které poskytuje WMI, se lze dotazovat pomocí WQL (Windows Management Instrumentation Query Language), což je podmnožinou SQL databázových jazyků. Tato vlastnost umožňuje využití výhod SQL, jako je například jednoduché filtrování získávaných informací nebo řazení podle zadaného kritéria [18].

WMI je primárně určená pro užití v aplikacích vytvořených v jazyce C nebo C++, navíc je nativně podporována frameworkem .NET, což zajišťuje její podporu u moderních vývojových nástrojů. Ovšem není problémem WMI využít i v jiných vývojových prostředích, která nejsou vybavena vestavěnou podporou této služby. V tomto případě plně postačuje, když jsou prostředí schopna přistupovat k ActiveX prvkům.

Služby nabízené WMI umožňují snadnou cestou získat velké množství detailních informací, ale cenou za tento komfort je vysoké využití systémových prostředků. Tato nepříjemná vlastnost omezuje jejich využití v oblasti monitorování systému v reálném čase, kdy je vhodnější využít standardních funkcí Windows API. Na druhou stranu nebrání žádná z okolností využití WMI pro získávání detailních doplňkových informací nebo informací, které nejsou pomocí standardních funkcí Windows API přístupné například kvůli systémem přiděleným oprávněním [18].

4.2.3 Klady a zápory používání služby WMI

Jak již bylo zmíněno v předchozích odstavcích, WMI je službou poskytující velké množství zajímavých informací, navíc velmi komfortní a bezpečnou cestou oproti klasickému řešení pomocí Windows API, ale jak již tomu bývá, každá mince má dvě strany. Cenou za tyto výhody je vyšší zatížení systému v momentě získávání požadovaných informací, proto je nezbytné důkladně zvážit při implementaci aplikací, jaké informace budou shromažďovány za pomoci WMI a pro které bude zvolena klasická cesta, aby nedocházelo ke zbytečnému zatížení zdrojů systému a k celkovému zpomalení jeho chodu. Další z drobných nevýhod WMI je její vlastnost, že je integrována jako součást systému a tak její rychlost závisí z velké části na celkové kondici systému Windows. Ovšem i přes tyto problémy je občas služby WMI vhodné a někdy i nezbytné využít. Příkladem mohou být úkony související se získáním informací týkajících se procesů, k nimž je potřeba disponovat neomezenými právy přístupu k systému. V současnosti je totiž snaha zvyšovat zabezpečení systému omezováním práv běžných aplikací, zvláště pak se tento trend projevuje u nejnovější verze operačního systému Windows 7.

5 Metody detekce anomálií v datech

Mezi základní problémy řešené systémy IDS patří schopnost rozlišení normálního a abnormálního chování uživatelů, aplikací a celých systémů. Právě k určení abnormálního chování se využívá shromažďování dat a jejich analýza pomocí různorodých metod detekce anomálií v datech. Tato kapitola textu se zabývá těmito metodami a popisem jejich základních principů, přičemž se zaměřuje na metody detekce anomálií, které jsou předmětem této práce. Anomálie detekované IDS systémy bývají ve většině případů statistické povahy, kdy modul obstarávající detekci vytváří modely chování zkoumaných subjektů, které porovnává pomocí metod detekce s aktuálním stavem. Modely chování subjektů bývají navrženy tak, aby postihovaly dobře jejich vnitřní strukturu a bylo je v průběhu času možno dobře aktualizovat. Pomocí detekce statistických anomálií jsou velmi snadno zjistitelné anomálie v datech reprezentovaných číselnou hodnotou, jako například počty vstupně/výstupních operací, počty přihlášení do systému, počty nových záznamů v logu za jednu hodinu atd.

5.1 Metriky metod detekce anomálií

Pro detekci abnormality je vhodné nalézt ukazatele, které mají dobrou vypovídací hodnotu o probíhajících činnostech. Vhodným příkladem sledovaných vlastností může být například hodnota reprezentující zatížení procesoru systému, počet otevřených síťových spojení nebo počet běžících procesů v systému, kdy se tyto hodnoty opakovaně měří v definovaných intervalech. Tyto hodnoty jsou pak funkcemi abnormality hodnot měření a jsou zahrnuty do vytvářeného profilu.

Obecně lze metody detekce anomálií rozdělit na dvě skupiny na základě kvantity jimi zkoumaných metrik.

První skupina se zaměřuje na detekci abnormality pomocí jednoho druhu metriky. Její hlavní výhodou je snadné zpracování dat a detekce anomálií, bohužel díky menší obecnosti klesají výrazně i vypovídající schopnosti této metody, neboť nejsou brány v úvahu vzájemné vztahy více monitorovaných vlastností systému.

Druhým typem jsou metody pracující na základě kombinace více metrik. Takto koncipované metody sledují najednou větší počet typů vlastností systému a mají pak komplexnější schopnosti detekce anomálií. Při kombinaci metrik jsou $S_1, S_2 \dots S_n$ hodnoty abnormality modelu $M_1, M_2 \dots M_n$. Vyšší hodnota S_i implikuje vyšší hodnotu abnormality. Základní kombinační funkci pro jednotlivá S můžeme pak definovat jako vážený součet čtverců 5.1.

$$a_1 S_1^2 + a_2 S_2^2 + \cdots + a_n S_n^2, \qquad a_i > 0 \tag{5.1}$$

kde a_i reflektuje relativní váhu metriky M_i. Metriky $M_1, M_2 \dots M_n$ nemusí být na sobě vzájemně nezávislé a tak nemusí užití této funkce poskytovat dobré výsledky, proto bývají definovány složitější komplexní funkce, založené například na Bayesovské pravděpodobnosti [8].

5.2 Bayesovská statistika

Jestliže $A_1, A_2 \dots A_n$ odpovídají jednotlivým měřením monitorovaných prostředků systému, mezi které lze řadit například míru využití operační paměti nebo využití procesoru, pak je možno na základě hodnot $A_1, A_2 \dots A_n$ rozhodnout pomocí Bayesovské statistiky, zda se jedná o anomálie nebo nikoliv. Každé A_i pak reprezentuje odlišný aspekt sledovaného systému a může nabývat dvou hodnot. V případě anomálie daného měření A_i = 1 a v případě, že se nejedná o anomálii, je A_i = 0 [8].

Buďiž H hypotézou, že systém byl narušen nebo v něm právě probíhá útok. Spolehlivost a citlivost detekce každého měření anomálie $A_1, A_2 \dots A_n$ je pak definována jako $P(A_i = 1|H)$ a $P(A_i = 1|\ \neg H)$. Pro ověření hypotézy H dosadíme hodnoty $A_1, A_2 \dots A_n$ do Bayesova teorému 5.2.

$$P(H|A_1, A_2 \dots A_n) = \frac{P(A_1, A_2 \dots A_n|H)\ P(H)}{P(A_1, A_2 \dots A_n)}$$

(5.2)

Toto bude vyžadovat společné pravděpodobnostní rozložení množiny měření podmíněných H a $\neg H$. Kvůli zjednodušení výpočtu předpokládejme, že jednotlivé A_i jsou na sobě vzájemně nezávislé a závisejí pouze podmíněně na hypotéze H a potom tedy: $P(A_1, A_2 \dots A_n|H) = \prod_{i=1}^{n} P(A_i|H)$ a $P(A_1, A_2 \dots A_n|\ \neg H) = \prod_{i=1}^{n} P(A_i|\ \neg H)$, což vede ke vztahu 5.3.

$$\frac{P(H|A_1, A_2 \dots A_n)}{P(\ \neg H|A_1, A_2 \dots A_n)} = \frac{P(H)}{P(\ \neg H)} \frac{\prod_{i=1}^{n} P(A_i|H)}{\prod_{i=1}^{n} P(A_i|\ \neg H)}$$

(5.3)

Toto znamená, že je možno určit pravděpodobnost, že došlo k narušení, danou jednotlivými hodnotami měření na základě předchozích pravděpodobností narušení za předpokladu, že se narušení vždy projeví jako anomální hodnota [8].

5.3 Predictive Pattern Generation

Prediktivní generování je způsob detekce průniků, který se snaží předpovídat události, jež budou s nejvyšší pravděpodobností následovat v budoucnosti, na základě událostí, ke kterým došlo již v minulosti. Metoda vychází z předpokladu, že pořadí akcí, ve kterém se provádějí, nebývá náhodné, ale vyznačuje se jistou pravidelností, kterou je možno definovat pomocí pravděpodobnosti výskytu události, následující po dokončení události právě probíhající. Pravidla, která popisují chování, bývají vytvářena induktivně, jsou dynamicky modifikována během učící fáze systému a lze je zapsat například v následující formě:

$$E1 - E2 - E3 \ \rightarrow \ (E4 = 60\%, E5 = 25\%, E6 = 15\%)$$

Toto pravidlo definuje, že událost E1 je následována událostí E2 a ta pokračuje událostí E3. Po dokončení události E3 může dojít k události E4 s pravděpodobností 60%, událost E5 bude následovat E3 s pravděpodobností 25% a nejméně pravděpodobnou událostí, která může E3 následovat je E6 s pravděpodobností 15%. Problémem metody prediktivního generování je nemožnost odhalit nebezpečnou aktivitu, která nebyla dříve popsána pravidly, v takovém případě je klasifikována jako neznámá. Neznámé aktivity je možno označit jako škodlivé, tím ale vzroste počet falešně pozitivních útoků nebo naopak neznámé aktivity označit jako neškodné, ovšem zde zase stoupne počet falešně negativně klasifikovaných útoků, které zůstanou mimo pozornost systému. V praxi pak bývají za nebezpečné aktivity označovány ty, které odpovídají levé straně pravidla, a pravá strana se velmi statisticky odchyluje od pravděpodobné následující události [20].

Mezi hlavní výhody metody prediktivního generování patří kromě celkové rychlosti algoritmu a možnosti jeho využití pro detekci probíhající v reálném čase fakt, že umožňuje na základě vzorů sekvencí odhalit útoky a nebezpečné aktivity, které by byly běžnými metodami neodhalitelné. Navíc tyto systémy bývají velmi dobře přizpůsobivé ke změnám chování uživatelů a aplikací v čase. Tato metoda je vhodná i pro monitoring chování uživatelů s velmi různorodým chováním, které se ale vyznačuje neměnnou sekvenčností. Systém brzy automaticky eliminuje nekvalitní pravidla a pro budoucí porovnávání zůstávají pouze pravidla kvalitní, které je možno využít vícenásobně a jsou platná po většinu času. Nespornou výhodou takto založených systémů je možnost detekce uživatelů,

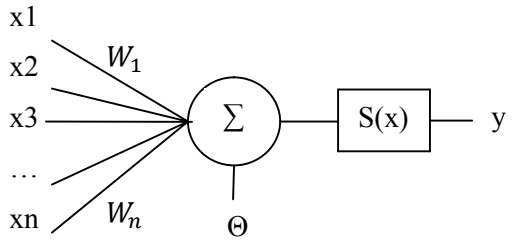
kteří se snaží ve fázi učení systému naučit systém špatné vzory chování. Děje se to uplatněním vestavěných, předem definovaných pravidel, v učící fázi [20].

5.4 Neuronové sítě

Neuronové sítě se začaly objevovat jako výpočetní model v oblasti umělé inteligence v období 40. let 20. století. Umělé neuronové sítě tvoří strukturu určenou pro distribuované paralelní zpracování dat. Neuronové sítě jsou složeny z umělých neuronů, které jsou vzájemně propojeny a předávají si signály, které transformují na základě přenosových funkcí.

5.4.1 Model umělého neuronu

Umělý neuron má jeden výstup a libovolný počet vstupů, opatřených vstupními vahami. Základní model umělého neuronu, který se v praxi běžně používá, byl popsán Pittsem a McCullochem:



Obrázek 5.1: Model umělého neuronu

V neuronu probíhají dva procesy, kterými jsou výpočet potenciálu a výpočet hodnoty výstupu pomocí aktivační funkce.

$$Y = S\left(\sum_{i=1}^{N}(w_i x_i) - \Theta\right) \quad (5.4)$$

- x_i jsou vstupy neuronu
- w_i jsou synaptické váhy
- Θ je práh
- S(x) je přenosová funkce neuronu
- Y je výstup neuronu

K výpočtu se používá několik druhů přenosových funkcí:

- Přenosová funkce radiální báze: $f(x) = e^{-kx^2}$
- Přenosová funkce hyperbolické tangenty: $f(x) = \frac{2}{1+e^{-kx}}$
- Sigmoidální přenosová funkce: $f(x) = \frac{1}{1+e^{-kx}}$
- Skoková přenosová funkce: $f(x) = 0$ pro $x < \Theta$ a $f(x) = 1$ pro $x \geq \Theta$

5.4.2 Klasifikace neuronových sítí

Umělé neuronové sítě je možno dělit podle způsobu jejich učení, kdy se odlišují dvě základní třídy, kterými jsou učení s učitelem a učení bez učitele. V závislosti na metodě učení je pak utvářena topologie vlastní neuronové sítě. Zde bude věnována pozornost několika základním typům

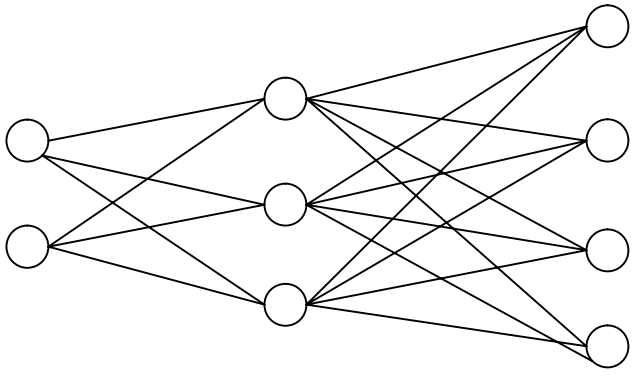
uspořádání. Pro správnou funkčnost neuronové sítě je klíčová trénovací množina vstupních vzorů, jež musí být správné velikosti a obsahovat pouze korektní vstupní vzory, jinak může dojít k přeučení sítě, případně jsou sítí naučeny i nekorektní vzory.

5.4.2.1 Učení s učitelem

Učení s učitelem je proces, kdy na vstup neuronové sítě vkládáme zadání problémů a síť produkuje na základě nastavení prahových úrovní a úrovní vah výstupní hodnotu na svém výstupu. Výsledek je porovnán s požadovaným řešením problému a je stanovena chyba. Následuje provedení korekce vah a prahů, aby byla chyba co nejvíce minimalizována. Tento proces se nazývá adaptace a je prováděn opakovaně, dokud není dosaženo minimální chyby.

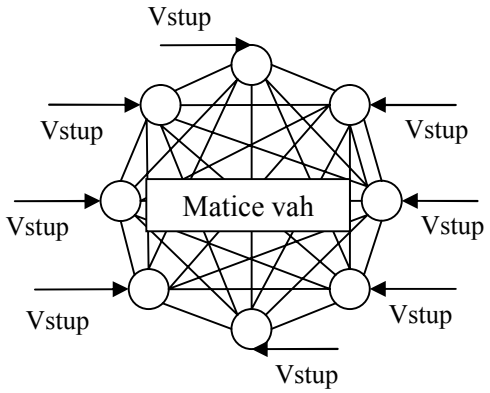
- **Perceptron:** jedná se o neuronovou síť složenou z jednoho neuronu, jejíž model je prezentován v předchozí části kapitoly. Neuronové sítě s touto topologií umožňují úspěšné řešení pouze lineárně separabilních problémů.

- **Vrstvená neuronová síť':** jedná se o sítě složené z více vrstev neuronů s dopřednými vazbami. Vstupy každého neuronu jedné vrstvy jsou napojeny na výstupy všech neuronů vrstvy předchozí. Každý neuron má tolik vstupů, kolik je v předchozí vrstvě neuronů. Jako přenosová funkce je často použita funkce sigmoidální. Při učení se používá různých chybových funkcí, z nichž nejčastější je kvadratická odchylka sítě, daná vztahem: $E(w) = \frac{1}{2}\sum_{i=1}^{n}(y_i - \hat{y}_i)^2$, kde y_i je hodnota cílového atributu a $\hat{y}_i$ je výsledek zařazení sítí. Pro učení neuronové sítě se velmi často používá algoritmus Back propagation neboli zpětné šíření chyby, jehož základem je předpoklad, že síť neobsahující žádné zpětné vazby a chyba se šíří zpět přes všechny vrstvy až k první vrstvě. Na základě zjištěné chyby se pak síť postupně adaptuje [21].



Obrázek 5.2: Vrstvená neuronová síť

- **Rekurentní neuronová síť:** příkladem rekurentních neuronových sítí může být Hopfieldův model, který používá symetrické spoje mezi neurony, kde propojení může být reprezentováno symetrickou maticí vah s nulovou hlavní diagonálou. Neurony se nacházejí buď v aktivovaném nebo neaktivovaném stavu. Postupným nastavováním vah dochází k zapomínání nepodstatných informací a naopak k posilováním vazby u důležitých informací. Rozdíl mezi touto topologií a klasickým vrstveným uspořádáním je, že klasické vrstvené sítě produkují výstupy ihned, ale Hopfieldovy sítě produkují výstupy až po určitém čase, kdy se dostanou do stabilního stavu.

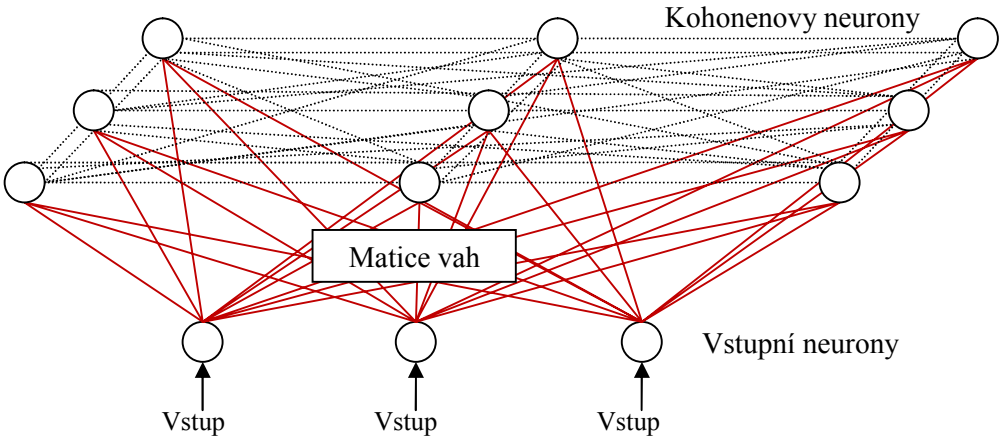


Obrázek 5.3: Hopfieldův model neuronové sítě

5.4.2.2 Učení bez učitele

Učení umělé neuronové sítě bez učitele je procesem, který není založen na vyhodnocování výstupu, ale naopak se tato metoda snaží vytvářet zobecněný výstup jen na základě požitých vstupních vzorů. Neuronová síť pracuje s neznámým obsahem vstupů, které si sama utřídí. Mezi nejčastější metody patří rozdělení podnětů do skupin podle podobnosti a určení typického zástupce pro každou skupinu. U učení bez učitele dochází dále ke klasifikaci na dlouhodobou paměť, tvořenou nastavením synaptických vah a paměť krátkodobou, tvořenou okamžitým stavem vzruchů.

- **Kohonenovy mapy:** reprezentují sítě tvořené vrstvou n vstupních neuronů a vrstvou tzv. Kohonenových neuronů, které jsou vzájemně propojeny každý s každým a zároveň spojeny se všemi vstupními neurony. Každému Kohonenovu neuronu náleží příslušný vektor synaptických vah, který je složen ze stejného počtu prvků jako vstupní vektor. Kohonenovy neurony se vzájemně ovlivňují a ustálí se ve stavu, kdy reaguje pouze ten neuron, který byl nejvíce excitovaný. Schopnosti sítě jsou dány počtem neuronů, které obsahuje. Principem funkce tohoto modelu je třídění vstupů do skupin, které si sama neuronová síť definuje [21]. Detailnímu popisu Kohonenových map a principů jejich činnosti se věnuje jedna z částí kapitoly o implementaci demonstrační aplikace, neboť právě Kohonenovy mapy byly zvoleny jako vhodný prostředek pro realizaci této aplikace. Pozornost je věnována zejména způsobu, kterým probíhá učení neuronové sítě, vyhodnocování výsledků a procesu předzpracování vstupních dat.



Obrázek 5.4: Kohonenoův model neuronové sítě

5.4.3 Využití neuronových sítí pro detekci anomálií

Neuronové sítě je možno využít v oblasti detekce anomálií v naměřených datech. Pro tento účel jsou zvláště vhodné vrstvené neuronové sítě a Kohonenovy mapy, které jsou popsány v předchozí části kapitoly. Typickým problémem detekce anomálií v oblasti systémů detekce průniků je obecné stanovení obvyklého chování systému. Neuronové sítě mají schopnost učit se z příkladů a schopnost abstrakce, což z nich dělá vhodné kandidáty pro řešení problémů v této oblasti. Díky schopnosti zobecnění jsou systémy založené na této bázi schopny detekovat i útoky nebo podezřelé aktivity, které jsou variantami již známých útoků nebo detekovat útoky podle typických doprovodných jevů a nemusejí tak být k úspěšné detekci s konkrétním útokem seznámeny. Vhodným příkladem může být analýza posloupností příkazů, které zadal uživatel během dne pomocí příkazové řádky systému. Monitorovány jsou četnosti zadávání příkazů a alokace systémových prostředků, nutná pro jejich vykonání. Neuronová síť, která je správně naučena typickému chování daného uživatele, je schopna rozlišit korektního uživatele od útočníka, který zadává jiné sekvence příkazů, případně příkazy, které mají za následek rozdílnou konzumaci systémových prostředků od profilu, kterému je síť naučena. V případě korektního uživatele je výstupem neuronu hodnota 1, v případě detekovaného útoku je hodnota rovna nule. Problémem této metody je, že se chování uživatelů v čase mění a proto je nutno profily přizpůsobovat [21].

5.5 Support Vector Machines

Suppor Vector Machines, neboli též algoritmy podpůrných vektorů, se řadí mezi metody strojového učení, hledající nadrovinu optimálně dělící trénovací data v prostoru příznaků, přičemž optimálnost rozdělení je definována jako maximum minima vzdálenosti mezi body rozdělovaných dat. SVM patří k relativně novým metodám a jsou založené na tzv. jádrových algoritmech, které se snaží o nalezení lineární hranice a zároveň se vyznačují schopností reprezentovat velmi složité nelineární funkce. V terminologii neuronových sítí je SVM síť s jednou skrytou vrstvou, tvořenou jádrovými jednotkami a s jednou prahovou výstupní jednotkou.

Hranice mezi třídami je pak $y = \sum_{i=1}^{N} w_i x_i + b$, kde x je vstup neuronu, w je váha a b je aktivační práh. Optimální separační hyperrovina pak maximalizuje rozpětí definované $\frac{|w^T x+b|}{\|w\|}$ a při dodržení podmínky $min|w^T x_i + b| = 1$ lze výraz definovat jako $\frac{1}{\|w\|^2} = \frac{1}{w^T w}$.

Určení hyperroviny je optimalizačním problémem a můžeme jej řešit Lagrangerovou metodou nedeterminovatelných koeficientů za užití vztahu 5.5.

$$L(w, b, \alpha_i) = \frac{1}{2} w^T w - \sum_i \alpha_i [y_i (w^T x_i + b) - 1] \quad (5.5)$$

kde $y_i = 1$, když x_i patří do jedné třídy a $y_i = -1$, jestliže patří do třídy druhé.

V současné době jsou jádrové metody předmětem intenzívního výzkumu jak v oblasti teorie, tak v oblasti aplikací. SVM se dívají na klasifikaci jako na kvadratický optimalizační problém a klasifikují data za pomoci stanovení souboru podpůrných vektorů, které náleží do množiny trénovacích dat. Oproti jiným, běžně používaným metodám, mezi které řadíme například neuronové sítě, nemusí docházet v případě SVM k redukci počtu vlastností kvůli problému přeučení, protože počet volných parametrů je závislý na hranici oddělující datové body a nikoliv na počtu vstupních parametrů. Díky těmto vlastnostem jsou SVM vhodné pro třídění širšího spektra problémů, přičemž jednou z hlavních pozitivních vlastností metody je binární klasifikace a regrese, které jsou důvodem nízké pravděpodobnosti zobecnění chyb. Díky rychlosti je možno metodu používat v reálném čase, výhodou je i schopnost pracovat s velkými soubory dat. Díky binární klasifikaci jsou data odpovídající běžnému chování označována hodnotou -1 a anomální data hodnotou 1 [25].

6 Demonstrační HIDS aplikace

Cílem praktické části této diplomové práce bylo na základě informací, získaných při studiu problematiky HIDS detekčních systémů, navrhnout a posléze implementovat vlastní HIDS aplikaci. Tato kapitola si klade za úkol popsat vytvořenou aplikaci a principy, na kterých je založena a které jsou využity k dosažení požadované funkcionality. Značná část kapitoly se zaměřuje na analýzu výsledků, které byly získány při testování aplikace v rámci různých režimů provozu na testovacím systému.

6.1 Specifikace zadání

Zadáním bylo specifikováno, že navrhovaná a posléze implementovaná aplikace má být systémem spadajícím do rodiny HIDS detekčních systémů, které pracují na principu detekce anomálií v naměřených datech. Tento požadavek byl klíčový pro volbu prostředků a metod, které se využívají k detekci útoků a abnormálních situací. Tento fakt také přesně definuje zdroje dat, které systém sleduje a analyzuje. Bližší informace o shromažďovaných datech, nezbytných pro úspěšnou detekci, jsou předmětem jedné z předchozích kapitol. Dalším z požadavků byl návrh systému jako řešení obsahujícího klientskou část, běžící na jednotlivých sledovaných výpočetních prostředcích a část serveru, jejímž primárním úkolem je vyhodnocování nashromážděných dat a jejich následná prezentace vhodným způsobem uživateli, v tomto případě spíše správci počítačových systémů.

6.2 Analýza problému

Prvním krokem při vytváření HIDS aplikace byla analýza problematiky HIDS systémů.

Na základě požadavků byla zvolena architektura aplikace klient – server, kdy mezi sebou tyto části komunikují pomocí komunikačního protokolu. Tento typ architektury je pro účel HIDS aplikace vhodný zejména kvůli dobré možnosti správy, kdy by v případě reálného provozu sytému bylo možno vyhodnocované informace sledovat z jednoho centrálního místa. Další nespornou výhodou, kterou s sebou toto řešení přináší, je minimalizace zatížení klientských stanic, kdy na nich dochází pouze ke shromažďování nezbytných údajů a výpočetně náročné vyhodnocování je v režii samostatné stanice, na které je provozována serverová část detekčního systému. Vedlejším efektem je také vyšší bezpečnost, neboť na klientských stanicích nemusí být ukládáno velké množství nashromážděných dat o chodu systému, která by mohla být nápomocna potencionálním útočníkům k lepšímu seznámení s konkrétním klientským systémem. Díky architektuře klient – server se také zvyšují možnosti přenositelnosti a kompatibility celého systému, kdy je v případě potřeby možno navrhnout a implementovat vlastní klientskou část, kompatibilní s požadovaným operačním systémem a komunikující na základě dále popsaného protokolu se serverovou částí, implementující detekční modul. V případě implementace klienta pro jiný systém je nutno uskutečnit pouze malé zásahy do serverové části, případně využívat pouze kompatibilní části detekčního modulu popsaného v dalších částech kapitoly. Jedná se zejména o specifické vlastnosti operačního systému, které nemohou být získány v systémech založených na jiné architektuře.

Vzhledem k zaměření na detekci pomocí odhalení anomálií v datech se nabízí celá řada možných metod, které jsou blíže popsány v předchozích kapitolách. Před samotným návrhem a implementací demonstrační aplikace bylo na malé množině dat testováno několik metod detekce

anomálií. Pro prvotní testování bylo využito softwarového nástroje NeuroSolution, pomocí kterého byla testována vhodnost přístupů založených na detekci anomálií pomocí klasifikátoru SVM, klasických vícevrstvých neuronových sítí s Backpropagation a Kohonenových samoorganizačních map. Jako množina dat, na která byly jednotlivé metody testovány, posloužila minutová statistika průměrných hodnot zatížení procesoru. V učící množině byla pouze data, která odpovídají normálnímu chování uživatele, naopak v testovací množině byla záměrně anomální data, lišící se více či méně od normálních hodnot. Cílem experimentu bylo seznámit se s detekčními vlastnostmi jednotlivých přístupů a determinovat vhodnou metodu pro budoucí demonstrační aplikaci. Nejlepších výsledků v případě tohoto nejzákladnějšího experimentu bylo dosaženo za použití Kohonenových map, kterým se blížila detekce za využití klasifikátoru SVM. Naopak nejslabší výsledek byl zaznamenán u klasické neuronové sítě, která se vyznačovala nejednoznačnou detekcí.

Pro experimentální aplikaci byla na základě studia problematiky HIDS a předchozích experimentů vybrána detekce anomálií pomocí Kohonenových map, které používají metodu učení bez učitele a jeví se vhodným kandidátem pro profilování chování uživatelů a systému.

6.3 Návrh a implementace aplikace

Demonstrační HIDS aplikace je tvořena dvěma částmi, které spolu vzájemně komunikují pomocí protokolu.

Pro běh klientské části aplikace bylo zvoleno prostředí operačního systému Microsoft Windows. Protože klientem shromažďované informace i metody jejich získání jsou specifické pro každou platformu, je klient závislý na konkrétní platformě. Pro realizaci klientské části bylo zvoleno vývojové prostředí Embarcadero Delphi, které se vyznačuje dobrými možnostmi manipulace se systémovými funkcemi a bezproblémovým přístupem k systémovým informacím. V rámci této práce byl vytvořen klient pro operační systémy Microsoft Windows, běžící na NT jádře. Jako základ klientské části aplikace byl zvolen a rozšířen nástroj pro monitorování systémových prostředků OS Windows [16]. Klient poskytoval pouze velmi základní funkcionalitu, potřebnou pro získávání dat pro potřeby HIDS. Jednalo se zejména o sledování zatížení procesoru a procesů běžících v rámci operačního systému. Pro jeho využití jako klientské části HIDS aplikace muselo být uskutečněno mnoho změn a musel být vytvořen nový modul, implementující sběr dat pro HIDS, moduly pro sledování přístupu do důležitých oblastí systému a modul obstarávající síťovou komunikaci se serverovou částí za využití vhodného aplikačního protokolu a zachování kompatibility. Detailnějšímu popisu návrhu a implementace tohoto klienta a zejména jeho částí úzce souvisejících s HIDS jsou věnovány některé z následujících úseků této kapitoly.

Serverová část aplikace implementuje vlastní jádro detekčního systému HIDS, které má za úkol zpracovávat a analyzovat zaznamenané charakteristicky chování uživatelů a počítačových systémů, jež jsou zasílány prostřednictvím jednotlivých klientů. Pro implementaci serverové části byl zvolen programovací jazyk Java a to zejména kvůli odstínění závislosti této části detekčního systému na konkrétní platformě. Serverová část kromě zpracování a vyhodnocování dat také informuje správce o stavu sledovaných stanic. Informace jsou reprezentovány jak ve formě textové tak i grafické pro jednotlivé části detekčního systému. Mezi hlavní operace této části HIDS aplikace patří získávání dat od jednotlivých klientů prostřednictvím síťového spojení za užití aplikačního protokolu, uchovávání shromážděných dat, učení se chování pro nově připojené stanice a vyhodnocování dat za pomoci naučených modelů chování v případě známých stanic. Většina kapitoly o implementaci demonstrační HIDS aplikace je věnována právě serverové části, zvláště pak detekčnímu modulu.

Aplikační protokol má za primární cíl umožnit komunikaci klientských stanic se serverovou částí. Jedná se o jednoduchý protokol, běžící na TCP/IP. Bezpečnost je zajištěna zapouzdřením pomocí SSL s využitím certifikátů. Detaily protokolu jsou prezentovány v samostatné části této kapitoly.

6.3.1 Použitá vývojová prostředí

Pro implementaci HIDS byla použita rozdílná vývojová prostředí a programovací jazyky pro klientskou a serverovou část. Pro serverovou část bylo zvoleno prostředí jazyka Java a pro klientskou část prostředí Embarcadero Delphi. Motivací k tomuto rozhodnutí byla snaha o co největší nezávislost serverové části na konkrétní platformě a naopak o úzkou vazbu mezi systémem a klientem, která poskytuje lepší možnosti přístupu k systémovým informacím. Dalšími argumenty pro volbu prostředí Delphi pro klientskou část aplikace byl fakt menší náročnosti na systémové prostředky než v případě Javy, menší zatěžování klientské stanice a dále také bezproblémový běh na většině operačních systémů Microsoft Windows bez nutnosti instalace podpůrných knihoven nebo frameworků.

6.3.1.1 Použité vývojové prostředí pro serverovou část HIDS

Pro serverovou část aplikace, která je hlavním objektem zájmu této práce, bylo zvoleno prostředí Java. K výběru Javy přispěly výše zmíněné okolnosti, z nichž největší váhu měla zejména nezávislost aplikací implementovaných v Javě na konkrétní platformě. Mezi velké výhody programovacího jazyka Java patří zejména široká celosvětová komunita, díky níž je zajištěna bohatá nabídka volně i komerčně šiřitelných knihoven pro nejrůznorodější oblasti použití a samozřejmě také silná podpora při řešení technických problémů ze strany zkušených expertů formou internetových diskuzích fór a skupin. Programovací jazyk Java a prostředí pro vývoj aplikací a jejich běh jsou v současnosti vyvíjeny společností SUN Microsystems.

Pro vývoj aplikací v prostředí jazyka Java existuje celá řada vývojových komerčních i volně šiřitelných vývojových prostředí. Mezi nejznámější zástupce prostředí šířených pod volnou licencí patří Netbeans a Eclipse. Pro demonstrační HIDS aplikaci bylo zvoleno propracované a uživatelsky přátelské prostředí NetBeans. V tomto vývojovém prostředí probíhala implementace objektového návrhu od prvních fází, kdy bylo implementováno jádro zajišťující funkcionalitu detektoru anomálií, až po poslední fáze, kdy bylo vytvářeno grafické uživatelské prostředí a důkladně testována funkčnost celého systému.

Vývoj probíhal na platformě Microsoft v operačním systému Windows 7 s 64 bitovým jádrem, následovalo pak testování serverové části na systémech Ubuntu Linux a Apple OS X Snow Leopard.

6.3.1.2 Použité vývojové prostředí pro klientskou část HIDS

Jako vývojové prostředí pro tvorbu klientské části demonstrační aplikace bylo s ohledem na výše popsané okolnosti zvoleno prostředí Embarcadero Delphi. Jedná se o prostředí již dříve vyvíjené společností Borland, v současnosti vyvíjené a nabízené společností Embarcadero. Embarcadero Delphi je propracované prostředí určené pro rychlou a efektivní tvorbu aplikací s grafickým uživatelským rozhraním pro operační systém Microsoft Windows. V komerční verzi je v současnosti nabízeno pod názvem “ Embarcadero Delphi 2010“ a ve verzi bez prémiového obsahu, určené pro volné využití, pod názvem “Turbo Delphi“. Hlavní rozdíly mezi volně šiřitelnou a komerční verzí jsou patrné z nabízené škály vestavěných komponent. Volně šiřitelná verze disponuje pouze několika

desítkami základních komponent, chybí pokročilé komponenty pro práci s databázemi, síťovými prostředky a další komponenty pro tvorbu specializovaných a pokročilých aplikací. Obě verze jsou kompatibilní s produktem Delphi, který dříve vyvíjela společnost Borland.
Vytvořená aplikace využívá pouze základních vestavěných komponent a je tedy bez problému přeložitelná pomocí volně šiřitelné verze prostředí, jako je například "Delphi 7 Personal", kde byla její funkčnost také plně testována [16].

Prostředí Delphi bylo zvoleno pro svoji výše zmiňovanou uživatelskou přívětivost, snadnost návrhu a možnosti tvorby efektivního uživatelského rozhraní vytvářené aplikace. K jeho volbě přispěl i fakt, že umožňuje bezproblémový přístup k funkcím operačního systému Windows pomocí vestavěných knihoven. Přístup k informacím nabízených skrze službu WMI je možno bez zásadních problémů realizovat pomocí schopnosti přistupovat k ActiveX prvkům [16].

Jako operační systém, na kterém byla aplikace vyvíjena, byl zvolen Microsoft Windows 7 v 64 bitové verzi.

6.3.2 Použité knihovny

Tato část kapitoly si bere za cíl seznámit čtenáře s knihovnami třetích stran, použitých při vývoji aplikace. Pro vývoj demonstrační HIDS aplikace bylo ve valné většině použito standardních knihoven a komponent, dodávaným společně s daným prostředím, byly ovšem použity i knihovny pro přidání zvláštní funkcionality. Všechny knihovny třetích stran použité v aplikaci jsou volně šiřitelné, přiložené u zdrojových kódů aplikace, případně získatelné ze zdrojů internetu.

6.3.2.1 Encog

Knihovna Encog je volně šiřitelnou knihovou pro práci s neuronovými sítěmi, jež obsahuje kromě široké škály topologií neuronových sítí také podpůrné funkce, umožňující snadnou manipulaci a operace s nimi. Implementuje celou řadu základních i pokročilých neuronových sítí, počínaje samostatným perceptronem, až po složité sítě typu samoorganizačních Kohonenových map.

Knihovna je určena pro jazyk Java a C# a je volně k dispozici na stránkách projektu v binární podobě i ve formě zdrojových kódů. K dispozici je spolu s knihovnou také demonstrační aplikace, umožňující práci se všemi druhy v knihovně podporovaných neuronových sítí, čímž může velmi dobře posloužit pro optimální výběr topologie sítě ve fázi analýzy problému.

V demonstrační HIDS aplikaci je knihovna použita v serverové části pro implementaci Kohonenových samoorganizačních map.

6.3.2.2 JFreeChart

Knihovna JFreeChart představuje volně šiřitelnou knihovnu pro jazyk Java, sloužící pro vytváření grafů v rámci grafického uživatelského prostředí a manipulaci s nimi. Pomocí této knihovny lze vytvářet široké spektrum grafů v nejrůznějších formátech.

Knihovna je použita pro vykreslování grafů, znázorňujících průběh detekce anomálií v serverové části demonstrační HIDS aplikace.

6.3.2.3 OpenSSL

OpenSSL představuje volně šiřitelné řešení, umožňující práci s protokoly TSL a SSL. Knihovna obsahuje bezpečnostní a kryptografické funkce pro zabezpečení TCP spojení. OpenSSL je multiplatformním řešením, které je použitelné v širokém spektru programovacích jazyků.

Knihovny OpenSSL jsou využívány klientskou částí demonstrační aplikace pro zajištění bezpečného SSL spojení se serverovou částí aplikace.

6.3.2.4 madCollection

Balík komponent madCollection obsahuje řadu knihoven a podpůrných funkcí pro manipulaci s prostředky operačního systému pro vývojové studio Delphi. Komponenty jsou rozděleny do několika oblastí dle jejich funkcionality. Jedná se o komponenty zaměřující se na práci s registrem operačního systému, systémovými voláními a operacemi s Windows API rozhraním. Balík komponent je šířen v podobě binárních souborů i zdrojových kódů.

Komponent z balíku madCollection je využito v klientské části aplikace, kde jsou potřebné pro monitorováná událostí souvisejících se souborovým systémem.

6.3.2.5 CoolTrayIcon

CoolTrayIcon je jednoduchou, volně šiřitelnou, grafickou komponentou pro vývojové prostředí Delphi. Jak je patrné již z názvu, jedná se o komponentu vytvářející ikonu v oblasti systémové lišty systému.

Komponenta je využita v rámci grafického uživatelského rozhraní klientské části aplikace.

6.3.2.6 ujGraph

Tato grafická komponenta je určena pro vývojové studio Delphi, kde slouží pro vytváření přehledných grafů. Komponenta je volně šiřitelná, obsahuje kompletní zdrojové kódy.

Komponenty ujGraph je použito v rámci grafického uživatelského rozhraní klientské části aplikace, kde jsou pomocí ní zobrazovány naměřené hodnoty.

6.4 Struktura demonstrační aplikace

Aplikace je navržena v rámci modelu klient – server a každá část je realizována formou samostatné aplikace. Následující úseky kapitoly popisují detailněji strukturu klientské a serverové části aplikace.

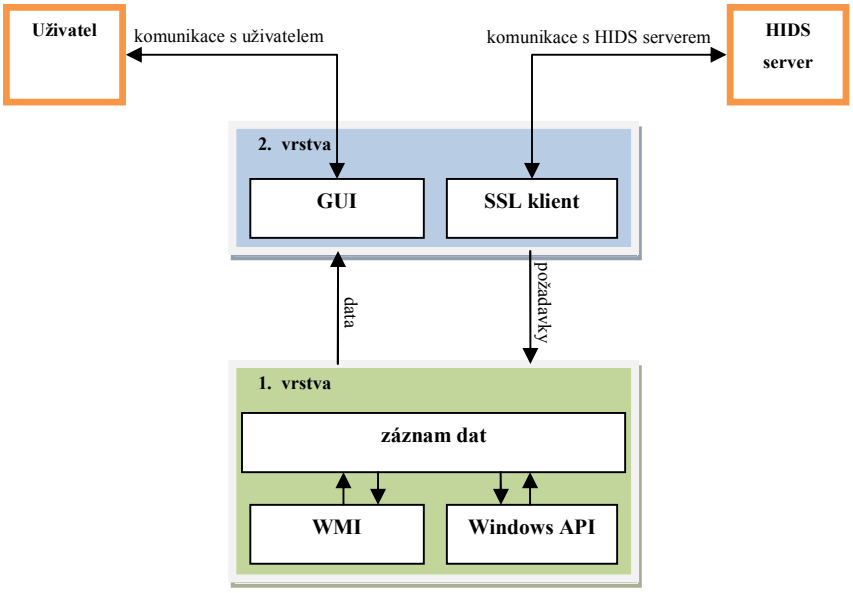
6.4.1 Struktura klientské části aplikace

Klientskou část aplikace lze z pohledu funkcionality rozdělit do dvou hlavních vrstev. Nižší vrstva poskytuje svoje služby vrstvě vyšší na základě požadavku. Vyšší vrstva je rozdělena na grafické uživatelské rozhraní a síťový modul, který zasílá naměřené údaje serverové části HIDS aplikace.

Úkolem nižší vrstvy je vytvoření nezávislého jádra běžícího na pozadí, jež slouží ke shromažďování informací o počítačovém systému a k jejich zpracování. Vyšší vrstva je pak tvořena grafickým uživatelským rozhraním, které má na starosti interakci s uživatelem a prezentaci získaných hodnot a síťovým modulem, obstarávajícím transport dat určených pro serverovou část HIDS aplikace.

Tento koncept předurčuje aplikaci k využití vláken, kdy jednotlivé vrstvy pracují nezávisle na sobě. Jejich využití je takřka nezbytné v případě nejnižší vrstvy, kdy je v některých případech použito

získávání informací pomocí služby WMI. Bez implementace vláken by nastala situace, kdy by byl značně snížen komfort ovládání uživatelského rozhraní, aplikace by přestala na několik okamžiků reagovat na podněty uživatele i systému a v krajním případě by mohlo dojít k omezení schopností jádra shromažďovat informace o systému. Tento fakt by mohl způsobit nekonzistenci monitorovaných dat a zkreslit značným způsobem celou statistiku [16].

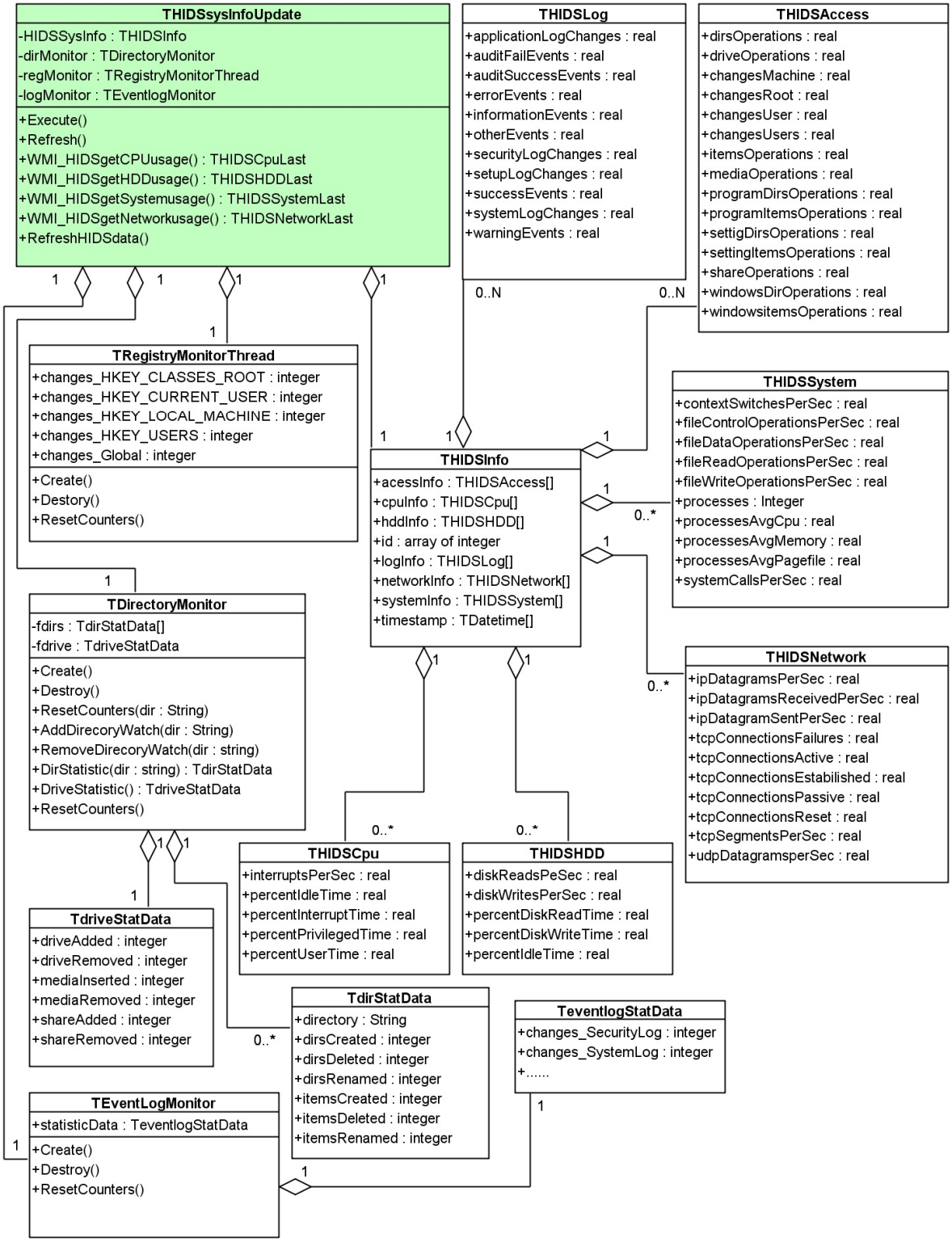


Obrázek 6.1: Vrstvený model klientské části

6.4.1.1 Jádro aplikace

Jádro aplikace je koncipované jako část programu běžící ve vlastním vlákně, která má na starosti získávání informací o monitorovaném systému za užití systémových funkcí rozhraní Windows API a služby WMI. Získávání informací probíhá periodicky. Každou sekundu jsou obnovovány údaje o prostředcích celého systému i jednotlivých běžících procesů. Následně po získání čerstvých informací dochází k aktualizacím statistik, jež uchovávají data pro pozdější využití [16]. V minutových intervalech dochází k vytvoření záznamu pro HIDS modul. Takto vytvořený záznam je pak předán vyšší vrstvě, která jej pomocí síťového modulu odešle serverové části HIDS aplikace, jež implementuje detekční logiku. Jádro je základním kamenem aplikace a proto je nezbytné zajistit jeho bezproblémovou funkčnost a dodržení periodičnosti časových intervalů, v nichž dochází k měřením.

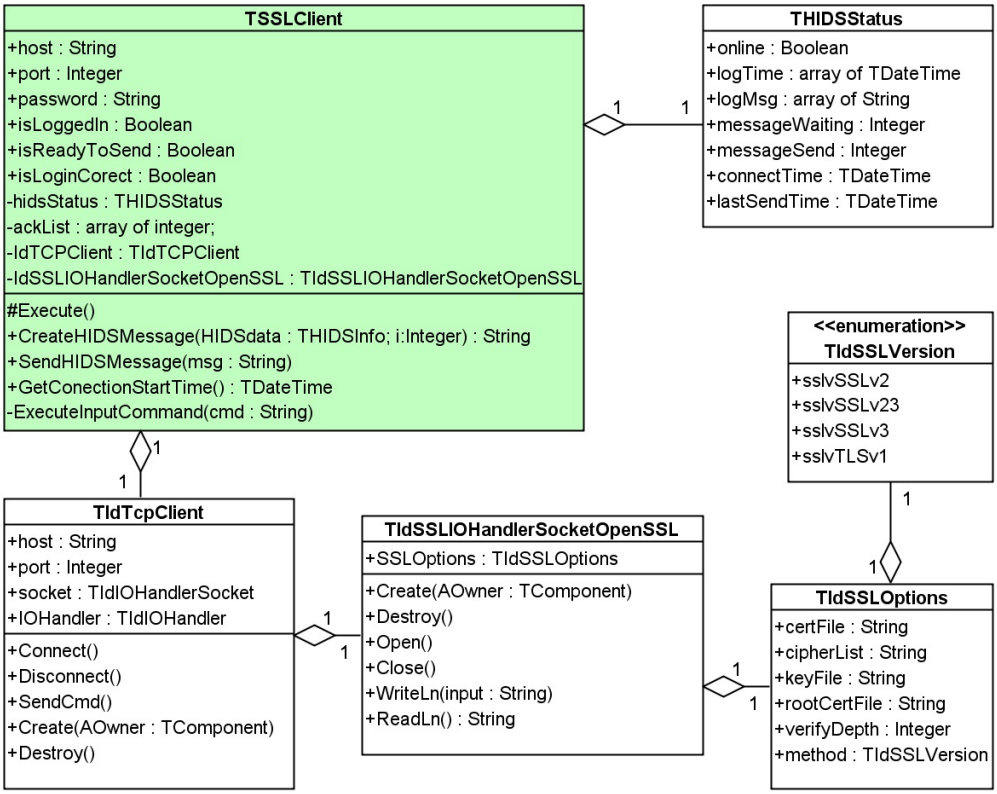
Na obrázku 6.2 je zobrazen diagram tříd jádra klientské aplikace. Nejdůležitější třídou z pohledu jádra je třída *THIDSsysInfoUpdate*, která implementuje vlákno, v jehož běhu dochází v periodických časových okamžicích ke sběru dat a jejich transformaci. Třídy *TRegistryMonitorThread*, *TDirectoryMonitor* a *TEventLogMonitor* poskytují této hlavní třídě svoje funkce, jejich prostřednictvím dochází k monitorování stavu registru, EventLogu a operacím se souborovým systémem. Struktura *THIDSInfo* slouží k uložení nashromážděných informací k pozdějšímu odeslání síťovým modulem.



Obrázek 6.2: Diagram tříd HIDS jádra klientské části aplikace

6.4.1.2 Síťový modul

Síťový modul implementuje funkcionalitu nutnou pro komunikaci se serverovou částí aplikace. Jeho úkolem je navázání a udržování spojení se serverovou částí a realizace reakcí na příchozí požadavky. Modul obsahuje prostředky pro konverzi zpráv do protokolem požadovaného tvaru a správu síťového spojení, zapouzdřenou pomocí bezpečnostního protokolu SSL. Hlavní činností modulu je periodické odesílání naměřených dat ve formě zpráv, jejichž formát je definován aplikačním protokolem. Síťový modul pracuje autonomně ve vlastním vlákně a vyřizuje požadavky nezávisle na zbytku klientské aplikace.



Obrázek 6.3: Diagram tříd síťového modulu klientské části aplikace

Diagram tříd na obrázku 6.3 prezentuje strukturu tříd síťového modulu klientské části aplikace. Hlavní třídu představuje *TSSLClient*, jež implementuje vlákno, které obstarává komunikaci se serverovou částí. Důležitou funkcí je *CreateHIDSMessage()*, jež transformuje nashromážděná data do formátu odpovídajícímu komunikačnímu protokolu aplikace.

6.4.1.3 Grafické uživatelské rozhraní

Uživatelské rozhraní monitorovací aplikace je navrženo se snahou dosáhnout dostatečné přehlednosti a uživatelské přívětivosti. Jeho hlavním úkolem je prezentovat naměřené údaje formou vhodného grafického výstupu uživateli a umožnit mu tak získání přehledu o sledovaném systému. Výstupy jsou realizovány pomocí grafů, které umožňují zobrazit monitorované údaje v různých časových úsecích. Tyto údaje jsou předzpracovány jádrem a jsou uživatelskému rozhraní předávány ve vhodném formátu k přímé prezentaci [16].

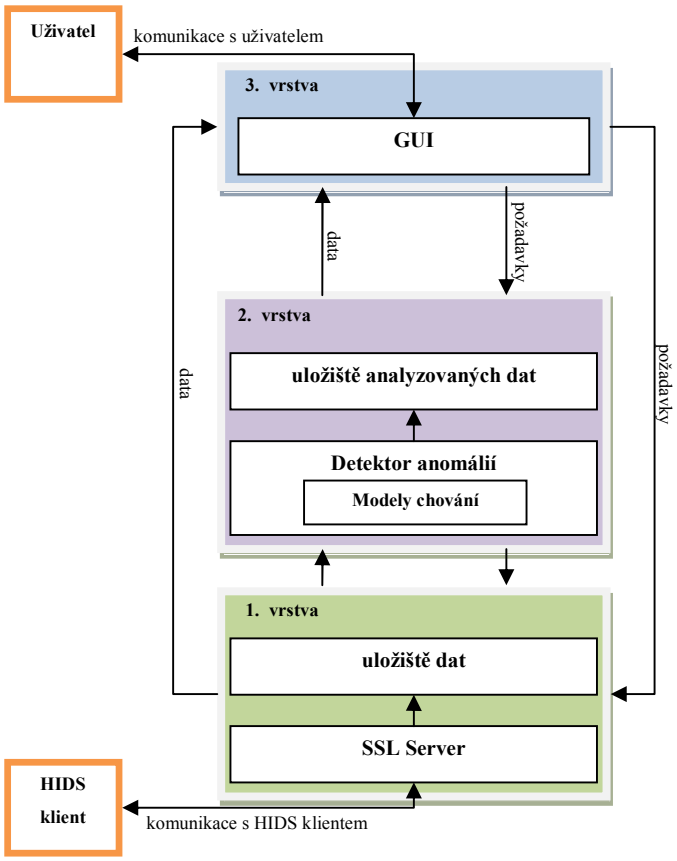
6.4.2 Struktura serverové části aplikace

Serverová část je navržena jako skupina vzájemně spolupracujících komponent, které je z hlediska funkcionality možno uspořádat do tří vrstev.

Nejnižší vrstva je zaměřena na získávání dat z jednotlivých klientských stanic, jejich následnou transformaci a uložení pro účely následného zpracování. Tato vrstva je tvořena síťovým modulem.

Střední vrstvu lze vnímat jako jádro aplikace, jehož činnost je autonomní a jehož primárním úkolem je zpracování získaných dat pomocí detektoru anomálií. Tato vrstva využívá služeb nižší vrstvy a poskytuje svoje služby nejvyšší vrstvě. Jádrem aplikace jsou implementovány operace učení detektoru a vyhodnocování nashromážděných dat v souladu s nastavením aplikace.

Nejvyšší vrstva je tvořena grafickým uživatelským rozhraním, jež má za úkol vhodným způsobem prezentovat analyzovaná data a v případě nutnosti upozornit uživatele na zjištěné nesrovnalosti.

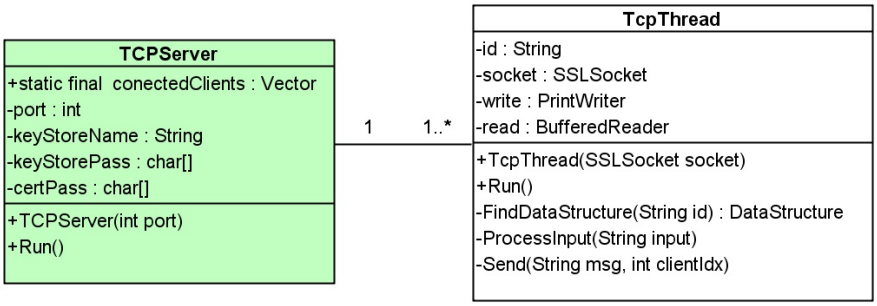


Obrázek 6.4: Vrstvený model serverové části

6.4.2.1 Síťový modul

Síťový modul lze považovat za nejnižší vrstvu serverové části demonstrační HIDS aplikace. Je tvořen konkurentním neblokujícím TCP serverem, který pracuje se zabezpečenými spojeními pomocí protokolu SSL s využitím certifikátů. TCP server umožňuje připojení a obsluhu více klientských stanic současně. Síťový modul je realizován v rámci samostatného vlákna aplikace a jeho účelem je zajištění komunikace s připojenými klienty na základě aplikačního protokolu, jež je definován v kapitole 6.5. Součástí modulu jsou nezbytné funkce pro komunikaci s klientskými stanicemi, zpracování příchozích zpráv, extrakci v nich obsažených dat a jejich následné uložení pro budoucí použití.

Na obrázku 6.2 je zobrazena třída *TCPServer,* implementující vlákno pro obsluhu síťového modulu. Třída *TCPThread* obsahuje metodu *ProcessInput(),* implementující převod přijatých zpráv ve formátu definovaném protokolem do formátu zpracovatelného HIDS aplikací,



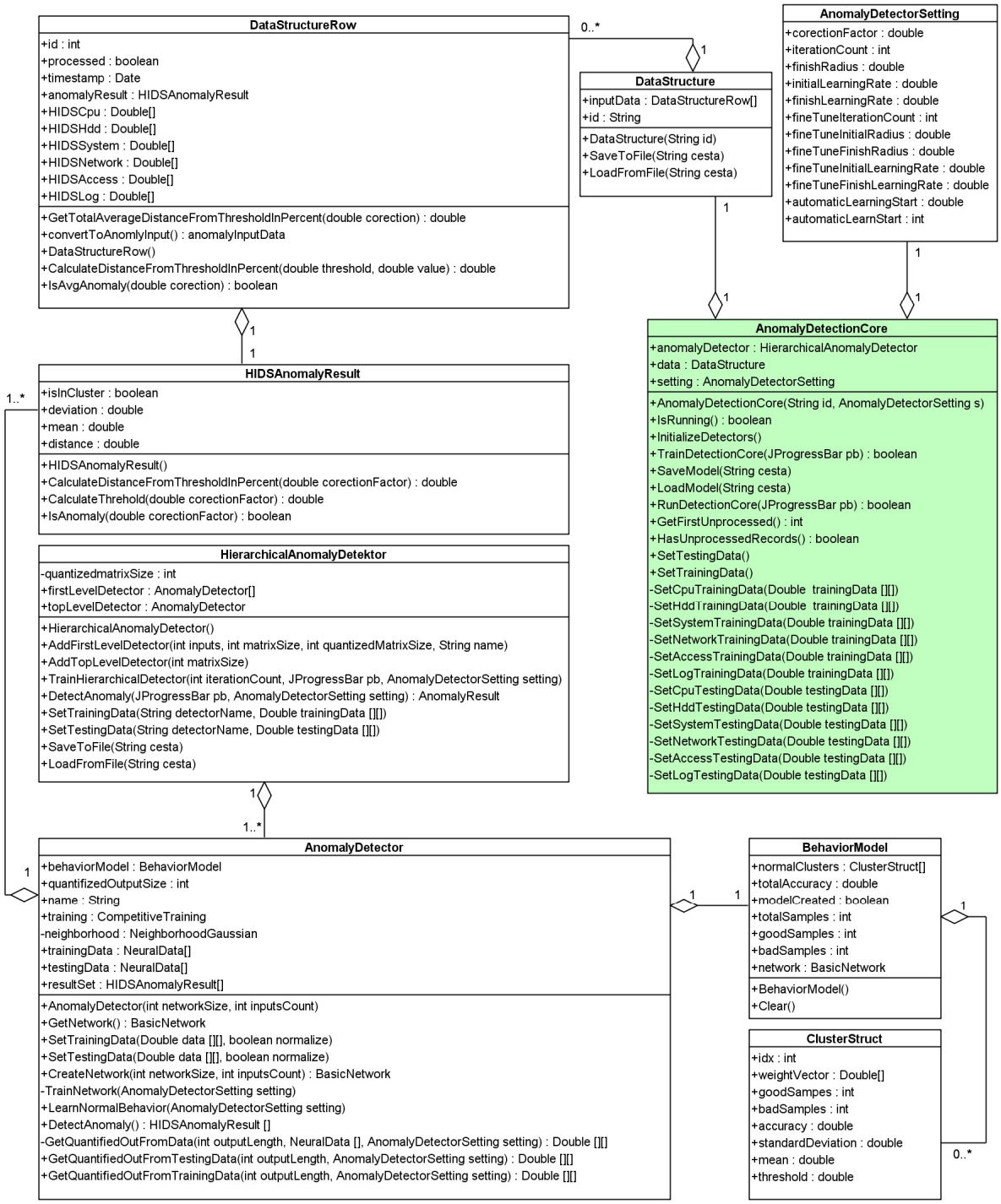
Obrázek 6.5: Diagram tříd síťového modulu serverové části aplikace

6.4.2.2 Jádro aplikace

Jádro aplikace je realizováno v samostatném vlákně a implementuje detektor anomálií. Detektor je složen ze skupiny detektorů nižší úrovně, které zpracovávají nashromážděné údaje z různých oblastí. Této problematice se věnuje velmi podrobně jedna z následujících částí této kapitoly.

Jádro má za úkol vykonávat dvě hlavní činnosti. První důležitou činností je učení se typickému chování v případě nově připojených klientských stanic. Druhou činností je vyhodnocování dat shromážděných z připojených stanic na základě jejich dříve vytvořených profilů.

Výsledkem činnosti jádra jsou zpracované statistiky, kdy je pro každý záznam určena hodnota anomality. Takto zpracované záznamy jsou pak bez problému použitelné pro prezentaci pomocí grafického uživatelského rozhraní.



Obrázek 6.6: Diagram tříd detekčního jádra serverové části aplikace

Na obrázku 6.6 jsou zobrazeny třídy detekčního jádra HIDS aplikace. Hlavní třídou je *AnomalyDetectionCore*, která obsahuje funkce pro provádění operací s jednotlivými dílčími detektory. Tato třída také zapouzdřuje strukturu nesoucí nastavení detekčního mechanismu a díky ní je realizováno i uložení vstupních dat pro zpracování. Podstatné jsou metody *TrainDetectionCore()* a *RunDetectionCore()*, které implementují profilování chování ze vstupních dat a vyhodnocení vstupních dat dle natrénovaného modelu. Třídy *HierarchicalAnomalyDetector* a *AnomalyDetector* pak implementují funkčnost jednotlivých detektorů.

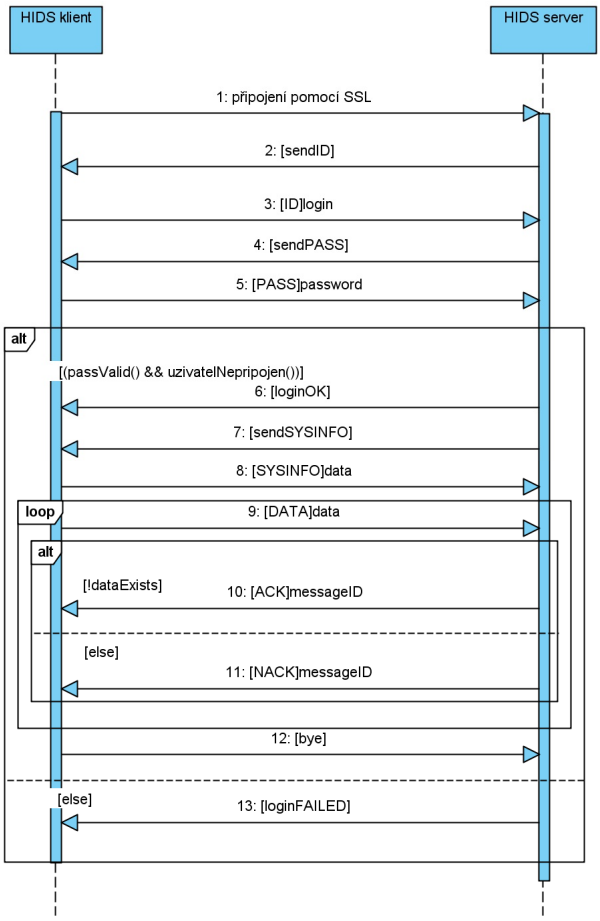
6.4.2.3 Grafické uživatelské rozhraní

Grafické uživatelské rozhraní demonstrační HIDS aplikace prezentuje grafickou i textovou formou výsledky analýzy nashromážděných dat pro jednotlivé klientské stanice. Ve formě tabulek jsou zobrazovány statistiky pro zvolenou stanici a na základě uživatelova požadavku je možno pro každý záznam zobrazit podrobné informace ve formě původních naměřených hodnot. Informace jsou doplněny grafickým výstupem, který přehledně formou grafu vizualizuje naměřené hodnoty, a jsou z něj jasně patrné okamžiky, ve kterých bylo detekováno chování odpovídající normálnímu a ve kterých naopak abnormálnímu chování pro daný profil chování. Podstatnou funkcí grafického uživatelského rozhraní kromě prezentace hodnot je také nastavení HIDS aplikace, manipulace se záznamy a operace s jednotlivými přihlášenými klientskými stanicemi.

6.5 Komunikační protokol

Vzhledem k použité architektuře klient – server bylo nutno navrhnout a posléze implementovat vlastní aplikační protokol, který by sloužil k přenosu dat mezi těmito částmi. Pro účely demonstrační aplikace je implementován tento protokol za použití protokolu TCP, spolu se zabezpečením pomocí bezpečnostního protokolu SSL.

Navržený protokol umožňuje základní operace autentizace klientů vůči serverové části aplikace a slouží především k přenosu naměřených dat směrem k serveru ve formátu strukturovaných textových zpráv. Tato část kapitoly se zaměřuje především na syntaxi příkazů protokolu a formát zpráv pro transport dat, dále se věnuje zabezpečení protokolu, které je nezbytné při manipulaci s citlivými daty.



Obrázek 6.7: Průběh činnosti komunikačního protokolu

V první fázi je úkolem komunikačního protokolu ověřit identitu uživatele. Jestliže proběhlo ověření uživatele v pořádku, následuje další fáze, ve které dochází k periodické výměně zpráv mezi klientem a serverem. Přijaté datové zprávy jsou serverem potvrzeny nebo odmítnuty a v případě potvrzení jsou serverovou částí uloženy a dále zpracovávány.

6.5.1 Příkazy protokolu

Aplikační protokol demonstrační aplikace využívá ke své činnosti několik řídících příkazů, které slouží zejména k ověření identity klientských stanic, zasílání požadavků na reakci partnerské aplikace a řízení navázaného spojení. Jednotlivé příkazy včetně popisu a příkladu použití jsou uvedeny v tabulce 6.1.

Příkaz	Popis
[sendID]	Požadavek serveru na identifikaci klienta zasílaný po zahájení spojení
[ID]*login_uzivatele*	Odpověď klienta na požadavek o identifikaci
[sendPASS]	Požadavek serveru na zaslání hesla
[PASS]*heslo_uzivatele*	Odpověď klienta na požadavek o zaslání hesla
[loginFAIL]	Sdělení serveru klientovi, že přístup byl zamítnut
[loginOK]	Sdělení serveru klientovi, že přístup byl povolen
[DATA]*datova_zprava*	Příkaz uvozující zprávu nesoucí data
[sendSYSINFO]	Požadavek serveru o zaslání systémových informací klientem
[SYSINFO]*systemove_informace*	Odpověď klienta na požadavek serveru o systémové informace
[bye]	Příkaz pro ukončení spojení mezi klientem a serverem
[ACK]id_zpravy	Pozitivní potvrzení datové zprávy
[NACK]id_zpravy	Negativní potvrzení datové zprávy

Tabulka 6.1: Popis příkazů HIDS aplikačního protokolu

6.5.2 Formát zpráv protokolu

Protokolem jsou k jeho činnosti používány dva základní typy zpráv. Prvním typem jsou zprávy obsahující informace o konfiguraci systému klienta, druhým, pro HIDS aplikaci důležitějším typem, jsou datové zprávy nesoucí naměřené údaje.

6.5.2.1 Formát informačních zpráv protokolu

Zprávy nesoucí informace o hardwarové a softwarové konfiguraci klienta jsou uvozeny příkazem "[SYSINFO]", za kterým následuje výčet systémových informací v přesně definovaném pořadí. Základní formát zprávy je následující:

[SYSINFO]S1;S2;S3;S4;S5;S6;S7;S8;S9;S10;S11

Tabulka 6.2: Základní formát zprávy nesoucí systémové informace

Jednotlivé parametry pak mají význam specifikovaný v tabulce 6.3.

Parametr	Popis parametru
S1	Název operačního systému
S2	Architektura operačního systému
S3	Registrovaný uživatel operačního systému
S4	Registrovaná organizace operačního systému
S5	Přihlášený uživatel
S6	Doména
S7	Název počítače

S8	Velikost fyzické paměti
S9	Počet jader procesoru
S10	Takt jádra procesoru
S11	Typ procesoru

Tabulka 6.3: Význam parametrů zprávy nesoucí systémové informace

Příklad skutečné zprávy, zaslané klientem serveru, je následující:

```
[SYSINFO]Microsoft Windows 7 Professional ;64;Ondra;;Ondra (Ondra-PC\Ondra);ONDRA-PC;WORKGROUP;4192760;4;2400;Intel(R) Core(TM)2 Quad CPU  Q6600  @ 2.40GHz
```

Tabulka 6.4: Příklad zprávy obsahující informace o klientském systému

6.5.2.2 Formát datových zpráv protokolu

Datové zprávy jsou prostředkem pro přenos dat mezi klientem a serverem. V rámci protokolu jsou uvozeny příkazem "[DATA]", za kterým následuje zpráva rozdělená do několika částí pomocí oddělovačů v přesně definovaném formátu, který musí být pro korektní činnost aplikace dodržen. Základní formát datové zprávy je následující:

```
[DATA][MSGID]id_zpravy|#|[TIMESTAMP]casove_razitko|#|[CPU]C1..C5|#|[HDD]H1..H5|#|[SYSTEM]S1..S10|#|[NETWORK]N1..N10|#|[ACCESS]A1..A15|#|[LOG]L1..L11
```

Tabulka 6.5: Základní formát datové zprávy

V datové zprávě odpovídají jednotlivé segmenty jednotlivým ukazatelům zatížení, které jsou nezbytné pro úspěšnou detekci anomálií v datech. V tabulce 6.6 jsou popsány dílčí parametry pro každý segment zprávy.

Segment	Parametr	Popis parametru
[CPU]	C1	Průměrný počet přerušení procesoru naměřený za 1 sekundu
	C2	Procentuální podíl nevyužitého procesorového času
	C3	Procentuální podíl času procesoru strávený v režimu přerušení
	C4	Procentuální podíl času procesoru strávený v privilegovaném režimu
	C5	Procentuální podíl času procesoru strávený v uživatelském režimu
[HDD]	H1	Procentuální podíl času pevných disků strávený čtením
	H2	Procentuální podíl času pevných disků strávený zápisem
	H3	Procentuální podíl nevyužitého času pevných disků
	H4	Průměrný počet zápisů provedených pevnými disky za 1 sekundu
	H5	Průměrný počet čtení provedených pevnými disky za 1 sekundu
[SYSTEM]	S1	Průměrný počet přepnutí kontextu za 1 sekundu
	S2	Průměrný počet řídících operací souborů za 1 sekundu
	S3	Průměrný počet datových operací souborů za 1 sekundu
	S4	Průměrný počet operací čtení ze souboru za 1 sekundu
	S5	Průměrný počet operací zápisu do souboru za 1 sekundu
	S6	Průměrný počet volání systému za 1 sekundu
	S7	Počet běžících procesů v systému
	S8	Průměrné zatížení CPU procesy
	S9	Průměrné využití operační paměti procesy
	S10	Průměrné využití stránkovacího souboru procesy

[NETWORK]	N1	Průměrný počet IP datagramů zpracovaných za 1 sekundu
	N2	Průměrný počet přijatých IP datagramů za 1 sekundu
	N3	Průměrný počet odeslaných IP datagramů za 1 sekundu
	N4	Počet navázaných TCP spojení
	N5	Počet TCP spojení zakončených chybou
	N6	Počet aktivních TCP spojení
	N7	Počet pasivních TCP spojení
	N8	Počet TCP spojení zakončených resetem
	N9	Průměrný počet TCP segmentů zpracovaných za 1 sekundu
	N10	Průměrný počet UDP datagramů zpracovaných za 1 sekundu
[ACCESS]	A1	Počet operací s disky
	A2	Počet operací s výměnnými médii
	A3	Počet operací se sdílenými složkami
	A4	Počet operací se složkami
	A5	Počet operací se soubory
	A6	Počet operací se složkami v systémovém adresáři
	A7	Počet operací se soubory v systémovém adresáři
	A8	Počet operací se složkami v programovém adresáři
	A9	Počet operací se soubory v programovém adresáři
	A10	Počet operací se složkami v adresáři dokumentů
	A11	Počet operací se soubory v adresáři dokumentů
	A12	Počet změn v ROOT klíči systémového registru
	A13	Počet změn v USER klíči systémového registru
	A14	Počet změn v MACHINE klíči systémového registru
	A15	Počet změn v USERS klíči systémového registru
[LOG]	L1	Počet změn v bezpečnostním logu operačního systému
	L2	Počet změn v systémovém logu operačního systému
	L3	Počet změn v instalačním logu operačního systému
	L4	Počet změn v aplikačním logu operačního systému
	L5	Počet nových záznamů logu operačního systému označených jako úspěch
	L6	Počet nových záznamů logu operačního systému označených jako chyba
	L7	Počet nových záznamů logu operačního systému označených jako varování
	L8	Počet nových záznamů logu operačního systému označených jako informace
	L9	Počet nových záznamů logu operačního systému označených jako úspěšný audit
	L10	Počet nových záznamů logu operačního systému označených jako neúspěšný audit
	L11	Počet nových záznamů logu operačního systému označených jako ostatní

Tabulka 6.6: Význam parametrů datové zprávy

Příklad skutečné zprávy, zaslané klientem serveru, je následující:

[DATA][MSGID]1304067183|#|[TIMESTAMP]06.05.2010|0:20|#|[CPU]6078.2107;68.7783;0.1483;5.0472;26.1641|#|[HDD]0.0944;0.1999;99.9464;7.5183;1.8176|#|[SYSTEM]16611.3460;790.0153;74.1698;53.9207;20.2492;1559674.5000;85.0000;0.3599;0.6751;0.6390|#|[NETWORK]165.4771;79.3668;86.1102;41.0000;0.0000;0.3802;0.0000;0.0000;171.1056;1.7686|#|[ACCESS]0.0000;0.0000;0.0000;0.0000;0.0000;0.0000;0.0000;0.0000;0.0000;0.0000;0.0000;7.0000;11.0000;7.0000|#|[LOG]0.0000;0.0000;0.0000;0.0000;0.0000;0.0000;0.0000;0.0000;0.0000;0.0000;0.0000

Tabulka 6.7: Příklad zprávy obsahující naměřená data

6.5.3 Zabezpečení protokolu

Důležitým aspektem, který musel být při návrhu tohoto protokolu brán v úvahu, je fakt, že aplikace provádí operace nad citlivými daty, které je nutno ochránit před potencionálními útočníky.

Cílem útočníků by mohlo být získání dat s motivem jejich využití pro zlepšení vlastní informovanosti o systému. Takto nashromážděné informace by pak v budoucnosti mohli využít pro napadení systému. Proti této hrozbě je protokol chráněn pomocí bezpečnostního protokolu SSL, který znemožňuje odposlech zpráv pomocí šifrování. Při připojení je pro serverovou část ověřena identita na základě certifikátů. Klientská část aplikace se prokazuje znalostí hesla, nutného pro vstup do systému. Protokolu SSL je ve stručnosti věnován úsek této kapitoly.

Další možnou hrozbu představuje snaha útočníků o skrývání nežádoucí aktivity na cílovém výpočetním systému. K maskování neautorizované aktivity by mohlo být použito například útoku přehráním, kdy by se útočník mohl pokoušet vkládat do spojení neaktuální, dříve naměřená data, jež by odpovídala normálnímu chování a skrývala tak před detekčním systémem skutečnou aktivitu. Pro omezení této hrozby jsou jednotlivé zprávy číslovány a jsou označeny časovým razítkem. Ze strany serveru jsou nově příchozí zprávy, které obsahují stejné časové razítko jako některá ze zpráv existujících, ignorovány.

6.5.3.1 Zabezpečení spojení pomocí SSL

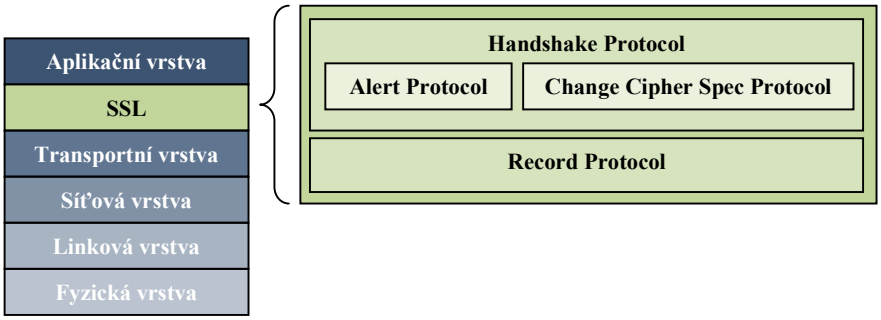
Pro zabezpečení aplikačního protokolu je demonstrační aplikací používáno služeb bezpečnostního protokolu SSL, implementovaného ve volně šiřitelné knihovně OpenSSL. Tato část kapitoly seznámí ve stručnosti čtenáře se základními vlastnostmi SSL protokolu.

Primárním cílem SSL protokolu je zajištění bezpečné komunikace a zajištění vhodné kombinace jiných protokolů pro šifrování, kompresi a autentizaci. SSL protokol zajišťuje spolehlivost a soukromí pro komunikující aplikace současně s ochranou jejich dat před odposloucháním a podvržením. Klíčovou vlastností je bezpečné šifrování, kdy po ustavení bezpečného spojení mezi dvěma komunikujícími uzly dojde k bezpečné výměně klíčů a následně pak je komunikace šifrována symetricky. Spolehlivost je zajištěna integrací Message Authentication Code v přenášených datech, kdy je tento bezpečnostní prvek používán pro kontrolu integrity zpráv. SSL je flexibilním řešením, které je platformě nezávislé a umožňuje vzájemnou komunikaci aplikací různých výrobců.

Důležitou vlastností je používání certifikátů pro komunikaci pomocí SSL. Certifikáty slouží zejména k ověření identity serverové části, aby klientská stanice měla jistotu, že nekomunikuje s podvrženým serverem, který se pokouší získat cenné informace. Pro účely SSL je možno používat certifikáty podepsané důvěryhodnou certifikační autoritou a takzvané self-signed certifikáty. V rámci certifikátu se vyskytují kromě podpisových informací také informace o jeho platnosti, veřejný klíč, Common name identifikující subjekt, pro který byl certifikát vydán a další informační údaje [2].

Protokol se v modelu ISO/OSI vyskytuje mezi transportní a aplikační vrstvou. Toto z něj dělá vhodného kandidáta pro ochranu protokolů, které nemají implementovány bezpečnostní funkce a

šifrování. Pro úpravu stávajících aplikací pak postačí přidání mezivrstvy, která bude obstarávat bezpečnost.



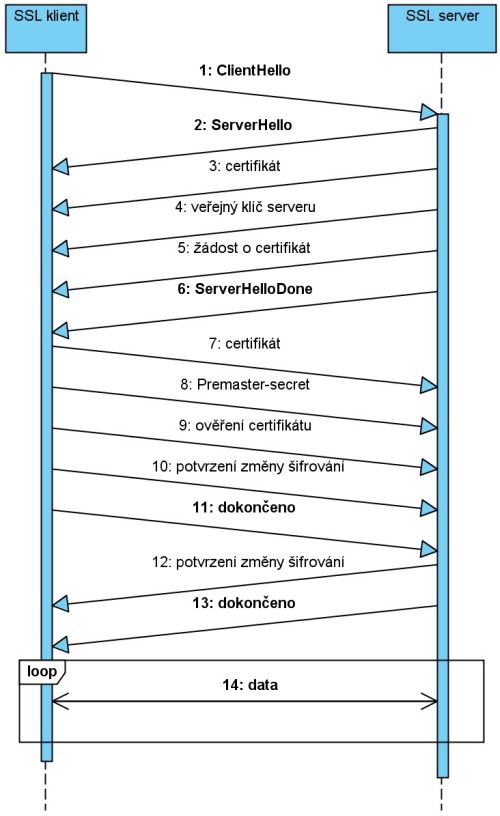
Obrázek 6.8: Umístění SSL protokolu v ISO/OSI modelu

SSL protokol lze obecně rozdělit do několika vrstev. Mezi dvě hlavní patří vrstva Handshake protokolu a vrstva Record protokolu.

Funkcí Record protokolu je připravovat data k odeslání a zpracovávat příchozí data. Data určená pro odeslání se fragmentují do jednotlivých bloků, následně mohou být komprimována a je pro ně vypočtena zabezpečující informace. Dalším krokem je šifrování dat a jejich předání nižší vrstvě.

Úkolem Handshake protokolu je navázání spojení spolu s identifikací komunikujících stran a nastavením potřebných parametrů. Protokol Handshake se dále skládá z Alert a Change Cipher Spec protokolů. Protokol při svojí činnosti přenáší informace sloužící k identifikaci spojení, determinaci způsobu komprese dat, informace o algoritmech šifrování a certifikáty pro ověření komunikujících stran.

Zjednodušený průběh navázání spojení a ověření komunikujících stran pak může vypadat následovně:



Obrázek 6.9: Navázání spojení mezi klientem a serverem pomocí SSL protokolu

Protokolem SSL jsou běžně používány blokové šifry RC2, DES, 3DES, IDEA, AES a z proudových šifer například RC4.

6.6 Detektor anomálií

Tento úsek kapitoly se hlouběji zabývá strukturou vlastního detektoru anomálií a principy, které jsou pro jeho činnost použity. Pozornost je věnována uspořádání detektoru, formátu vstupních dat, procesům učení a také problematice vyhodnocení výstupních hodnot.

Anomální chování uživatelů, aplikací a celého systému, které neodpovídá modelu standardního chování, je v demonstrační aplikaci detekováno detektorem, založeným na bázi Kohonenových samoorganizačních map. Tento typ neuronové sítě byl stručně popsán již v předchozích kapitolách, součástí této kapitoly je pak detailnější popis způsobu učení a vyhodnocování dat sítí.

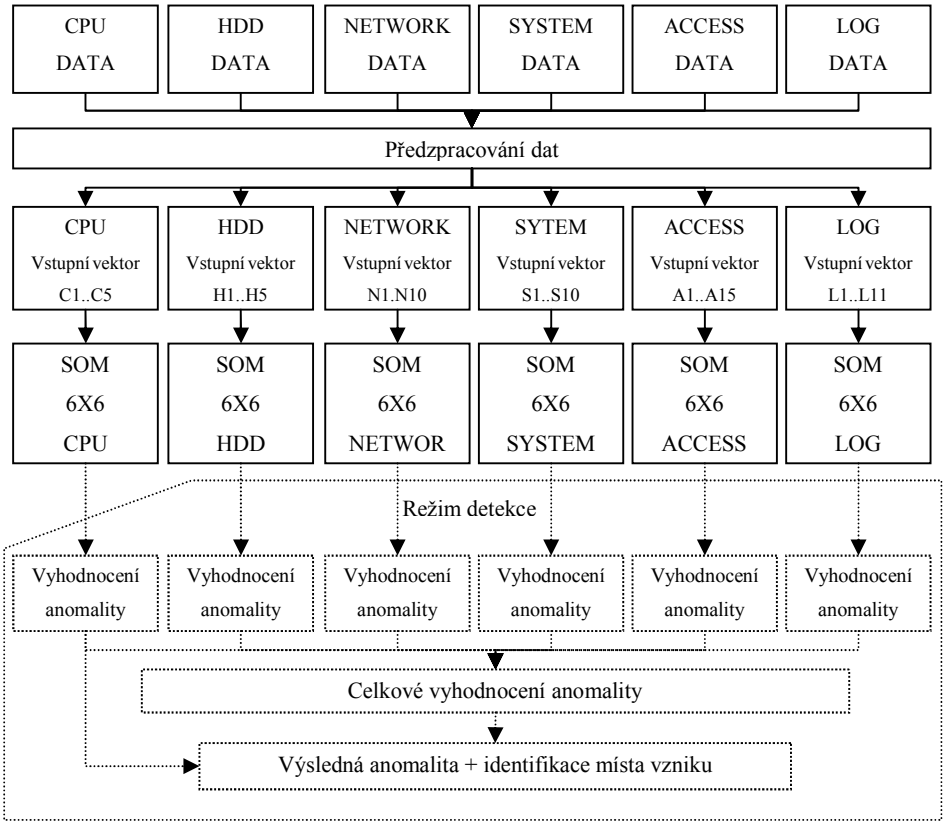
Kohonenovy mapy byly zvoleny jako vhodný přístup na základě studia dané problematiky a následného experimentálního testování na malé sérii vzorových dat. Využívají se k segmentaci a v případě detektoru anomálií produkují jako svůj výsledek rozdělení mapy na oblasti odpovídající normálnímu chování, vytvořené na základě trénovací množiny dat a ostatní oblasti, odpovídající abnormálnímu chování.

Činnost detektoru lze rozdělit na dvě základní aktivity. Prvním typem aktivity je učení se běžnému chování a druhým typem pak porovnávání aktuálního chování s vytvořeným profilem a na základě analýzy určení, zda se jedná o situaci spadající do normálního modelu chování nebo o situaci abnormální.

Pro potřeby demonstrační HIDS aplikace bylo využito volně šiřitelné knihovny neuronových sítí **Encog,** která implementuje Kohonenovy samoorganizační mapy a proceduru jejich učení. Pro úplnost je nutno způsob, jímž je proces učení a vyhodnocování implementován, popsat v této části kapitoly.

6.6.1 Struktura detektoru anomálií

Detektor anomálií, použitý v demonstrační aplikaci, je detektorem složeným z několika samostatných Kohonenových map. V modelu se vyskytuje celkem 6 základních Kohonenových map o velikosti 6x6 neuronů ve výstupní vrstvě, kdy je každá mapa určena pro zpracování jedné konkrétní oblasti dat, získaných z klientské části. Jednotlivé mapy mají tedy na starosti zpracování statistických údajů o procesoru, pevném disku, síťových spojeních, systémových prostředcích, přístupech k systémovým oblastem a logu operačního systému. Všechny tyto Kohonenovy mapy se na sobě nezávisle učí a samostatně vyhodnocují svůj úsek naměřených dat z testovací množiny.



Obrázek 6.10: Struktura detektoru anomálií

Každá Kohonenova mapa tedy zpracovává normalizovaná a předzpracovaná data v podobě vstupního vektoru a v případě, že se jedná o režim učení, produkuje model nebo v režimu detekce určuje hodnotu anomality daného vzorku.

Výsledkem je profilování chování a možnost detekce pro každou konkrétní skupinu dat bez závislosti na ostatních skupinách. Takto zpracovaná data jedné části detektoru ovšem nejsou v kontextu s vyhodnocenými daty detektorů ostatních. Proto je nutné data dále zpracovat, aby mohla být určena celková hodnota anomality, která poskytuje globálnější pohled na výsledek detekce. Bylo experimentováno s několika přístupy a uspořádáními struktury detektorů. Výsledkům tohoto experimentování a finálnímu řešení se podrobně věnuje část této kapitoly, zaměřující se na způsoby celkové vyhodnocování analyzovaných dat.

6.6.2 Vstupní hodnoty a jejich předzpracování

Hodnoty získané prostřednictvím klientské části aplikace je nutno vhodným způsobem předzpracovat. Pro použití Kohonenových map je nezbytné, aby vstupní hodnoty byly normalizovány a umístěny do intervalu <-1,1> pro dosažení optimálních výsledků.

Prvním krokem je normalizace hodnot. Tento krok je důležitý, neboť hodnota některých parametrů je reprezentována velmi malými číselnými hodnotami, naopak některé parametry dosahují velkých hodnot. Pro Kohonenovu samoorganizační mapu musí hodnoty co nejlépe pokrývat vymezený prostor hodnot, jinak dochází k degradaci výsledku. Pro normalizaci se jako optimální ukázal vztah [25]:

$$nv[i] = \frac{v[i]}{\sqrt{\sum_K v[k]^2}} \tag{6.1}$$

kde nv[i] představuje výslednou normalizovanou hodnotu parametru s indexem i, v[i] představuje původní hodnotu parametru s indexem i, K reprezentuje počet parametrů obsažených ve vektoru. Po aplikaci normalizace nabývají všechny prvky vektoru hodnoty z intervalu <0,1>.

Následným krokem, nutným pro optimální detekční schopnosti Kohonenovy neuronové sítě, je umístění hodnot normalizovaného vektoru do rozsahu <-1,1>. Pro tento účel je možno využít jednoduchého výpočtu dle vztahu:

$$ov[i] = (nv[i] * 2) - 1 \tag{6.2}$$

kde ov[i] reprezentuje normalizovanou hodnotu umístěnou do intervalu <-1,1> a nv[i] je normalizovanou hodnotou v intervalu <0,1> získanou z předchozího vztahu.

6.6.3 Režim učení chování uživatele

Pro úspěšnou detekci anomálií musí být vytvořen model, který profiluje standardní chování uživatele. Tento model vzniká v režimu učení, kdy jsou jednotlivým Kohonenovým samoorganizačním mapám dodávána trénovací data, na jejichž základě dochází k postupným úpravám vah sítě.

6.6.3.1 Učení Kohonenovy neuronové sítě

Pro proces učení je použito takzvané soutěžní strategie, kdy spolu neurony obsažené ve výstupní vrstvě vzájemně soupeří o to, který z nich bude aktivní. Přitom platí, že v jednom časovém okamžiku může být aktivní pouze jeden z neuronů a ten je pak také označován jako vítězný. Další důležitou vlastností sítě tohoto typu je, že po stanovení vítězného neuronu dochází ke změně vah jak samotného vítězného neuronu, tak i vah neuronů v jeho okolí na základě použité funkce okolí.

Soutěžení výstupních neuronů spočívá ve výpočtu vzdálenosti vektoru vah každého neuronu od vstupního vektoru dat. Jako vítězný neuron je vybrán neuron s nejnižší hodnotou výstupu a v následujících vztazích je označován jako index c. Vítězný neuron lze určit dle následujícího vztahu:

$$c(x) = arg_i min \|x(t) - w_i\| \tag{6.3}$$

Pro výpočet vzdáleností mezi vektory se obvykle používá Euklidovské vzdálenosti. Po nalezení vítězného neuronu následuje modifikace jeho vektoru vah a dále i váhových vektorů neuronů v jeho okolí. Modifikace vah okolních neuronů je důležitá pro vylepšení jejich pozice vůči novému

tréninkovému vzoru. Velikost okolí, jež bude modifikováno, je závislá na použité funkci. Základní funkce okolí je definovaná jako:

$$h_{ci}(t) = h(\|r_c - r_i\|; t) \tag{6.4}$$

kde r_c je pozicí vítězného neuronu, r_i je pozicí i-tého neuronu ve výstupní vrstvě Kohonenovy mapy. V demonstrační aplikaci je použita Gaussova funkce okolí, která se lépe hodí pro účely analýzy dat. Gaussova funkce je definována vztahem:

$$h_{ci}(t) = \begin{cases} \alpha(t) exp\left(-\frac{\|r_c - r_i\|^2}{2R^2(t)}\right), \|r_c - r_i\| < R(t) \\ 0, jinak \end{cases} \tag{6.5}$$

kde R značí poloměr okolí, $\alpha(t)$ je pak parametrem učení. Tyto parametry postupně s přibývajícími iteracemi učícího cyklu klesají. Během procesu učení dochází k modifikaci vektorů vah okolních neuronů dle pravidla:

$$w_i(t+1) = w_i(t) + h_{ci}(t)[x(t) - w_i(t)] \tag{6.6}$$

kde x(t) je vstupní hodnotou v čase t a $h_{ci}(t)$ je funkcí okolí se středem umístěným na pozici vítězného neuronu c ve výstupní mapě neuronů [26].

Učení probíhá v mnoha iteracích, kdy jsou data opakovaně umisťována na vstup Kohonenovy neuronové sítě. Je vhodné pro zvýšení efektivity rozdělit proces učení na dvě fáze, kdy v první proběhne rychlé přerozdělení vah sítě a v druhé s větším počtem iterací dochází k jemnému doladění a ustálení hodnot vah.

6.6.3.2 Vytvoření profilu chování

Dalším krokem, který následuje po naučení Kohonenovy samoorganizační sítě, je vytvoření modelu chování uživatele, který pak bude spolu s natrénovanou sítí sloužit k detekci anomálií. Model chování je vytvářen na základě množiny trénovacích dat, jimž je již síť naučena.

Prvním krokem při tvorbě modelu normálního chování je určení shluků odpovídajících normálnímu chování a určení prahových hodnot pro každý cluster. Prahové hodnoty se následně používají k rozhodnutí, zda vstupní vektor ještě náleží do clusteru reprezentujícího normální chování nebo již překračuje práh a bude označen jak abnormální. Seznam clusterů normálního chování je vytvořen z trénovací množiny dat, kdy je do seznamu přidán index neuronu s nejmenší vzdáleností ke vstupnímu vektoru za předpokladu, že se v seznamu ještě nevyskytuje. Nejbližší neuron výstupní vrstvy ke vstupnímu vektoru je neuron, mezi jehož vektorem vah a vstupním vektorem je nejmenší vzdálenost:

$$c_{normal}(x) = arg_i min\{d(x(t), w_i)\} \tag{6.7}$$

kde c_{normal} je indexem neuronu výstupní vrstvy, značícího cluster normálního chování, $d()$ je vzdálenost mezi vstupním vektorem x(t) a neuronem výstupní vrstvy w_i.

Pro výpočet vzdálenosti slouží Euklidovská vzdálenost, kterou je možno obecně vyjádřit dle vztahu 6.8.

$$d(p,q) = \sqrt{(p_1 - q_1)^2 + (p_2 - q_2)^2 + \cdots + (p_n - q_n)^2} = \sqrt{\sum_{i=1}^{n} (p_i - q_i)^2}$$

(6.8)

kde p, q jsou body a n je počet dimenzí prostoru.

Druhým krokem profilování normálního chování je určení prahu pro každý normální cluster. Prahovou hodnotu by bylo možno definovat konstantně pro celý model, ale jako vhodnější způsob se jeví její určení na základě výpočtu součtu standardní odchylky a průměru vzdáleností vstupních vektorů vůči náležejícím clusterům normálního chování. Tato metoda je založena na předpokladu, že vzdálenosti mezi normálními clustery a vstupními vektory vykazují normální rozložení a většina hodnot je právě menší než součet jejich standardní odchylky a průměru. Prahovou hodnotu lze vyjádřit vztahem:

$$T = (\mu + (c * \sigma))$$

(6.9)

kde μ je standardní odchylkou a σ je průměrem hodnot vzdálenosti vstupních vektorů od příslušných clusterů normálního chování. Parametr c je pak korekčním faktorem, umožňujícím zvýšit nebo snížit hodnotu prahu [26].

Proces tvorby modelu normálního chování je možno popsat následovně pomocí pseudokódu:

1. Vektory vah pro každý neuron výstupní vrstvy:
   $W = \{w_1, w_2, \dots, w_a, .., w_p\}$
2. Vytvoření seznamu váhových vektorů normálních clusterů:
   $W_{normal} = \{w_k, w_l, w_m, .., w_q\}, kde\ q \leq p$
3. For $w_i \in W_{normal}$ do
   - Výpočet vzdáleností: $d(v_{ji}, w_i)$, kde v_{ji} je vstupní vektor náležející clusteru normálního chování s vektorem vah w_i
   - Výpočet průměru vzdáleností μ_i a standardní odchylky vzdáleností σ_i

   End For;

Seznam indexů clusterů normálního chování spolu s prahovými hodnotami a natrénovaným modelem sítě tvoří model normálního chování. Tento model je vytvořen pro každou ze šesti Kohoenových map a je uložen pro budoucí detekci.

6.6.4 Režim detekce anomálního chování

Druhou hlavní funkcí detektoru anomálií je na základě vytvořených profilů chování určit, zda analyzovaná data odpovídají abnormálnímu nebo normálnímu chování. Proces vyhodnocování testovacích dat využívá ke své činnosti seznamu clusterů, odpovídajících normálnímu chování a vypočtených prahových hodnot pro tyto clustery.

Vyhodnocování probíhá na základě určení příslušnosti vstupního vektoru ke clusteru, který je opět definován neuronem výstupní vrstvy Kohonenovy mapy. Tento neuron je neuronem s nejmenší vzdáleností od vstupního vektoru. K výpočtu je použit opět vztah:

$$c_{normal}(x) = arg_i min\{d(x(t), w_i)\} \quad (6.10)$$

kde je vzdálenost d () Euklidovskou vzdáleností.

V následujícím kroku je ověřeno, zda přiřazený cluster je obsažen v seznamu clusterů normálního chování. V případě, že se v tomto seznamu cluster nevyskytuje, je zde proces detekce pro daný vzorek ukončen a vzorek je určen jako abnormální a zcela se lišící od vzorků vykazujících normální chování.

V případě výskytu clusteru v seznamu clusterů normálního chování je vypočtena vzdálenost vstupního vektoru od středu clusteru za použití Euklidovské vzdálenosti. Za předpokladu, že je vypočtená vzdálenost menší než prahová hodnota příslušného clusteru normálního chování, je vzorek označen jako normální, v opačném případě je označen za vzorek vyznačující se anomálií.

Proces detekce anomálního chování je opět možno popsat pomocí pseudokódu [26]:

1. x= vstupní vektor
2. For $w_i \in W\ do$
   - Výpočet vzdáleností: $d(x, w_i)$

   End For;
3. $d_x = \min d(x, w_i)$
4. $w_x = arg^i \min d(x, w_i)$, kde w_x^i je vektor vah náležící clusteru s nejmenší vzdáleností vůči vstupnímu vektoru
5. IF $w_x \notin W_{normal}$ then
   - x je označeno jako abnormální
6. Else IF $d_x \leq T$, kde T je prahová hodnota pro daný cluster normálního chování
   - x je označeno jako normální
7. Else
   - x je označeno jako abnormální

Tento postup je aplikován na každou z šestice Kohonenových map a pro každou je tak určeno, zda část dat jí zpracovávaná vykazuje známky abnormálního či normálního chování, případně jak moc je toto chování vzdáleno od prahu normálního chování.

6.6.5 Výstupní hodnoty a jejich vyhodnocení

Posledním krokem činnosti detektoru anomálií je celkové vyhodnocení daného vzorku. V předchozích krocích došlo k vyhodnocení jednotlivých částí pomocí šesti samostatných Kohonenových map. Výstupem této fáze jsou vzdálenosti od prahů normálního chování pro jednotlivé Kohonenovy mapy a na jejich základě vytvoření informace o jejich příslušnosti k normálním nebo abnormálním vzorům chování. Cílem celkového vyhodnocení je na základě těchto částečných výsledků určit, zda vzorek jako celek vykazuje normální či abnormální chování, případně určit jeho vzdálenost od prahu normálního chování. Pro celkové vyhodnocení vzorků byly testovány dva rozdílné přístupy, které budou na tomto místě prezentovány.

6.6.5.1 Vyhodnocení na základě průměrné hodnoty jednotlivých Kohonenových map

První způsob určení výsledku celkové detekce představuje jednodušší řešení, které se ukázalo při testování jako efektivnější a proto vhodnější pro konečnou implementaci. Pro jednotlivé Kohonenovy mapy lze v percentuální podobě určit vzdálenost jejich vstupních vektorů od prahu normálního chování pro odpovídající cluster dle následujícího jednoduchého vztahu:

$$D_x = -\frac{100}{T} * (T - d_x)$$

(6.11)

kde D_x značí vzdálenost v procentech od prahu T, d_x je pak vzdáleností vstupního vektoru od vektoru vah příslušného clusteru. Kladné výsledné hodnoty indikují abnormální chování, naopak záporné hodnoty odpovídají chování normálnímu.

Celkový výsledek detekce v percentuální podobě, udávající vzdálenost od prahu normálního chování pak lze určit dle vztahu:

$$D_{total_x} = \frac{D_{CPU_x} + D_{HDD_x} + D_{NETWORK_x} + D_{SYSTEM_x} + D_{ACCESS_x} + D_{LOG_x}}{6}$$

(6.12)

kde kodnota D_{total_x} vyjadřuje celkovou průměrnou vzdálenost od prahu normálního chování, v případě záporné hodnoty indikuje normální chování a naopak kladnou hodnotou je reprezentováno abnormální chování.

6.6.5.2 Vyhodnocení na základě Kohonenovy mapy vyšší úrovně

Druhá z metod, vedoucí k určení celkového výsledku, byla realizována přidáním Kohonenovy samoorganizační mapy vyšší úrovně, obsahující 10x10 neuronů ve výstupní vrstvě. Tato síť na svém vstupu zpracovává vektor tvořený z hodnot odpovídajících vzdálenostem vstupních vektorů od jednotlivých neuronů obsažených ve výstupní vrstvě jednotlivých Kohonenových map nižší úrovně. V případě použití šestice map nižší úrovně pracuje Kohonenova mapa vyšší úrovně se vstupním vektorem o délce 216 prvků, což je výpočetně velmi náročné a neefektivní. Možnou cestou, nabízející redukci ze 36 výstupů na pouhých 6 pro každou Kohonenovu mapu, představuje výpočet funkčního potenciálu prezentovaný v [27]. Výpočet funkčního potenciálu pro každou mapu probíhá ve čtyřech krocích:

1. Zjištění potenciálu pro každý neuron výstupní mapy sítě dle vztahu:

$$P_t(w(j)) = \sum_{i=1}^{M} \exp\left(-\alpha \|w(i) - w(j)\|^2\right)$$

(6.13)

   kde w(j) značí neuron výstupní vrstvy Kohonenovy mapy, $P_t(w(j))$ reprezentuje potenciál, M počet neuronů výstupní vrstvy, α je poloměr clusteru.
2. Výběr neuronu s nejvyšším potenciálem a jeho přidání do seznamu výsledných clusterů, tento neuron je odstraněn ze seznamu kandidátních neuronů a nebude dále zpracováván.
3. Potenciál všech neuronů je snížen o hodnotu potenciálu nalezeného v bodě 2 dle vztahu:

$$P_{t+1}\big(w(j)\big) = P_t\big(w(j)\big) - P_t(\mathrm{w}^*)\exp\left(-\beta \|w(i) - w(j)\|^2\right)$$

(6.14)

   kde t+1 je indexem potenciálu neuronu, který bude modifikován, w^* je neuronem asociovaným se středem clusteru a β je poloměr clusteru, kdy platí že $\beta < \alpha$.

4. Návrat ke kroku 2, pokud není vybráno 6 neuronů s nejvyšším potenciálem.

Pro každou mapu jsou hodnoty potenciálu normalizovány dle vztahu 6.15.

$$y = \frac{1}{1 + \|w - x\|} \tag{6.15}$$

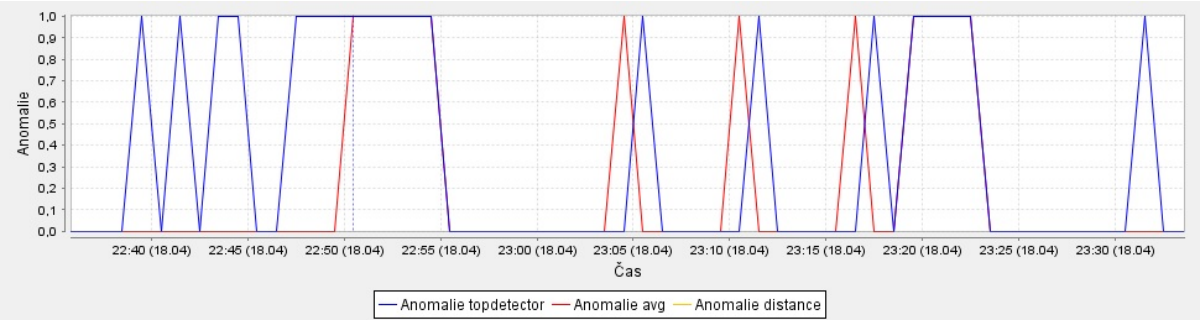
kde w značí střed clusteru a x je originálním vstupním vektorem.

Takto získané hodnoty lze použít jako vstupní vektor Kohonenovy mapy vyšší úrovně a následně na ni aplikovat dříve zmiňovaný algoritmus učení a detekce.

6.6.5.3 Porovnání metod celkového vyhodnocování

Ačkoli se přístup používající Kohonenovu mapu vyšší úrovně jevil jako nadějný, na základě experimentů byl upřednostněn jednodušší postup, používající ke své činnosti výpočet průměrné hodnoty anomálií jednotlivých Kohonenových map. Výsledky detektoru vyšší úrovně byly v některých případech neurčité a často se projevovaly výskytem falešně pozitivních nebo naopak falešně negativních výstupů modelu, proto byl tento přístup použit pouze ke srovnání možných metod.

V následujícím krátkém testu byly prováděny nestandardní operace, na jejichž provádění nebyl profil chování vytvořen. Ze zobrazených hodnot grafu jsou patrné odchylky obou prezentovaných metod.



Obrázek 6.11: Porovnání metod celkového vyhodnocení detektoru

Začátek	Konec	Aktivita	Anomálie	Detektor vyšší úrovně	Průměr detektorů nižší úrovně
22:36	22:38	Žádná aktivita	NE	NE	NE
22:39	22:39	Spuštění prohlížeče	NE	NE	NE
22:40	22:49	Prohlížení webových stránek	NE	ANO	NE
22:50	22:56	Kopírování 2GB dat	ANO	ANO	ANO
22:57	23:10	Žádná aktivita	NE	NE	NE
23:11	23:12	Mazání souborů na HDD	ANO	ANO	ANO
23:13	23:16	Žádná aktivita	NE	NE	NE
23:16	23:17	Čištění registru systému	ANO	NE	ANO
23:18	23:19	Žádná aktivita	NE	NE	NE
23:21	23:23	Defragmentace disku	ANO	ANO	ANO
23:24	23:30	Žádná aktivita	NE	NE	NE

Tabulka 6.8: Popis testovacích dat pro graf porovnání celkového vyhodnocení detektoru

Na výsledku tohoto krátkého testu je vidět vyšší přesnost metody založené na průměrování, na větší sérii dat jsou pak odchylky obou metod zřetelnější a rozdíly mezi nimi se prohlubují.

6.7 Grafické uživatelské rozhraní aplikace

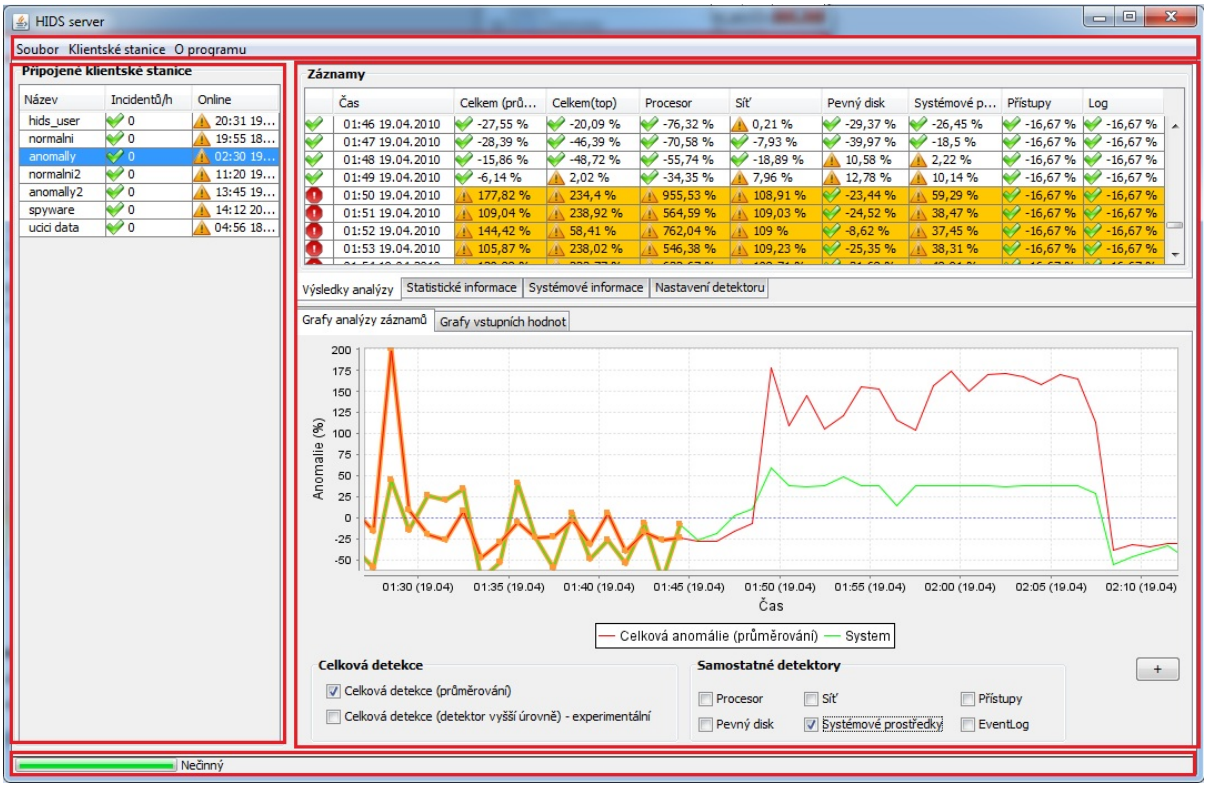
Grafické rozhraní aplikace bylo vytvořeno s důrazem na jednoduchost a dobré možnosti ovládání uživateli. Grafická uživatelská rozhraní jak klientské, tak i serverové části, jsou realizována s použitím standardních komponent a knihoven pro tvorbu grafů. Na základě testování byla rozhraní obou částí upravována, aby bylo dosaženo co nejlepší ovladatelnosti a přehlednosti.

6.7.1 Grafické uživatelské rozhraní serverové části

Serverová část aplikace nabízí přehledné uživatelské rozhraní, sloužící k získání přehledu o jednotlivých připojených klientských stanicích a stavu analyzovaných dat. Prostřednictvím uživatelského rozhraní lze prohlížet analyzované záznamy jednotlivých stanic a provádět nastavení, jež ovlivňují činnost detekčního modulu aplikace.

6.7.1.1 Hlavní okno aplikace

Veškerá interakce serverové aplikace s uživatelem probíhá prostřednictvím hlavního okna aplikace. Toto okno lze z pohledu funkcionality rozdělit na několik částí, jejichž funkce je popsána v následujícím úseku textu.



Obrázek 6.12: Hlavní okno serverové části aplikace

6.7.1.2 Aplikační menu

Ve vrchní části okna HIDS aplikace se nachází menu, prostřednictvím kterého je možno přistupovat k jednotlivým funkcím aplikace. Z aplikační nabídky jsou přístupné operace pro manipulaci s datovými modely detekčního systému a přístup k uživatelským nastavením serverové části aplikace. Prostřednictvím této nabídky je také možno přistupovat k funkcím exportování a importování dat a funkcím umožňujícím správu profilů klientských stanic.

6.7.1.3 Seznam připojených klientů

V levé části hlavního okna je zobrazen seznam připojených klientských stanic spolu se základními informacemi o jejich názvu, počtu detekovaných anomálií za poslední hodinu a časového okamžiku, kdy byla obdržena poslední datová zpráva. Kliknutím pravým tlačítkem na položku seznamu lze vyvolat kontextové menu, jež nabízí možnosti manipulace se shromážděnými vzorky, natrénovanými modely i profily klientských stanic.

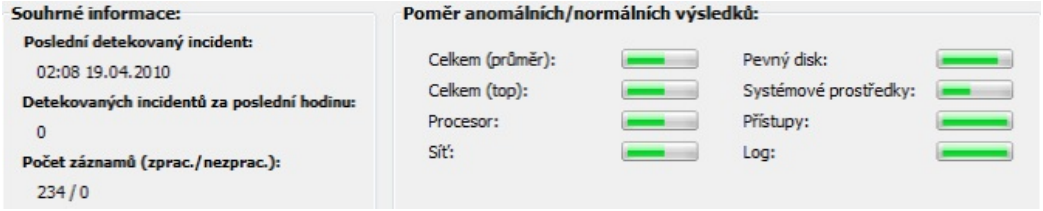
6.7.1.4 Textová reprezentace analyzovaných dat a informace o klientské stanici

Ve vrchní polovině střední části okna se nacházejí panely spolu přepínačem, jehož prostřednictvím lze zobrazit výpis analyzovaných dat ve formě tabulky, kde jsou přehledně prezentovány celkové hodnoty anomality i hodnoty anomality zjištěné dílčími částmi detektoru pro jednotlivé záznamy. Hodnotou anomality se chápe vzdálenost od prahu normálního chování v percentuálním formátu a v případě že, přesahuje hranici 0%, je tento fakt v tabulce viditelně indikován. Tabulka je propojena s panely ve spodní části okna, obstarávajícími grafickou reprezentaci analyzovaných dat a zobrazení detailních informací o vybraném záznamu. Informace zobrazené v těchto panelech jsou závislé na vybraném prvku v této textové tabulce. Těmto panelům je věnována v další části této kapitoly.

	Čas	Celkem (průměr)	Celkem(top)	Procesor	Síť	Pevný disk	Systémové...	Přístupy	Log
!	22:54 18.04.201...	101,49 %	56,72 %	616,93 %	-0,17 %	-11,88 %	37,39 %	-16,67 %	-16,67 %
!	22:55 18.04.201...	108,63 %	9223372...	350,99 %	-0,58 %	313,51 %	21,2 %	-16,67 %	-16,67 %
!	22:56 18.04.201...	69,14 %	-76,49 %	-56,31 %	13,41 %	342,43 %	148,66 %	-16,67 %	-16,67 %
✓	22:57 18.04.201...	-11,23 %	-75,46 %	6,28 %	-5,25 %	-16,67 %	-18,39 %	-16,67 %	-16,67 %
✓	22:58 18.04.201...	-4,33 %	-75,78 %	-9,15 %	13,41 %	-56,23 %	59,32 %	-16,67 %	-16,67 %
✓	22:59 18.04.201...	-22,68 %	-66,09 %	-44,21 %	9,59 %	-58,18 %	-9,93 %	-16,67 %	-16,67 %

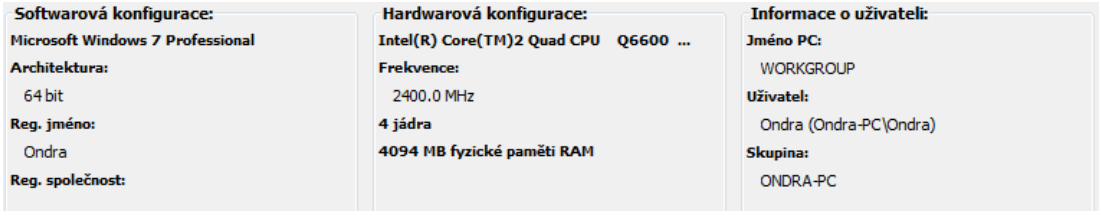
Obrázek 6.13: Výpis analyzovaných dat

Druhým přepínačem v pořadí se aktivuje statistický přehled informací pro konkrétního vybraného klienta. V panelu jsou zobrazeny informace o počtech detekovaných anomálií a grafickou formou jsou zobrazeny poměry normálních a anomálních záznamů, detekovaných dílčími částmi detektoru.



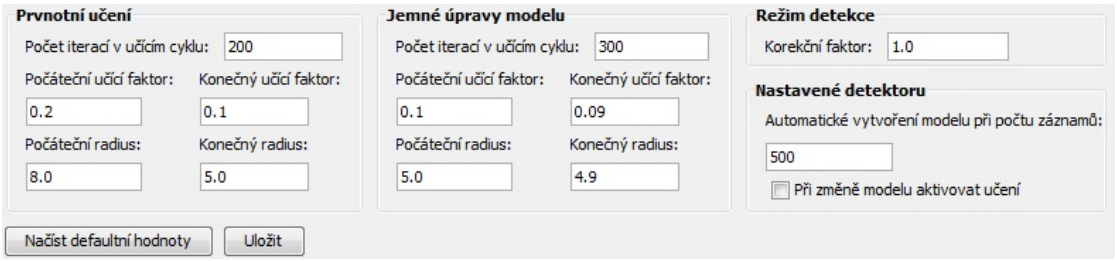
Obrázek 6.14: Statistický přehled informací o klientovi

Užitím třetího přepínače je zobrazen panel, obsahující informace o základní hardwarové a softwarové konfiguraci klientské stanice.



Obrázek 6.15: Informace o hardwarové a softwarové konfiguraci klienta

Prostřednictvím čtvrtého přepínače je možno vyvolat nastavení pro individuální klientskou stanici. Je možno nastavovat parametry učení detekčního jádra, korekční faktor či množství záznamů, při kterém se aktivuje automatické vytvoření profilu.

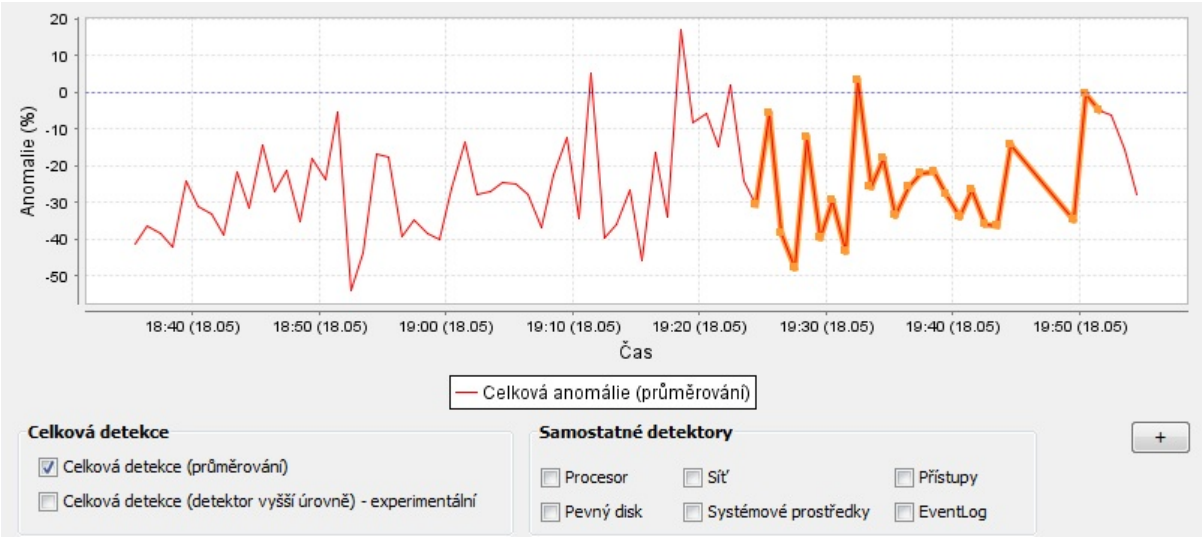


Obrázek 6.16: Konfigurace detektoru anomálií

6.7.1.5 Grafická reprezentace analyzovaných dat a detailní informace o záznamech

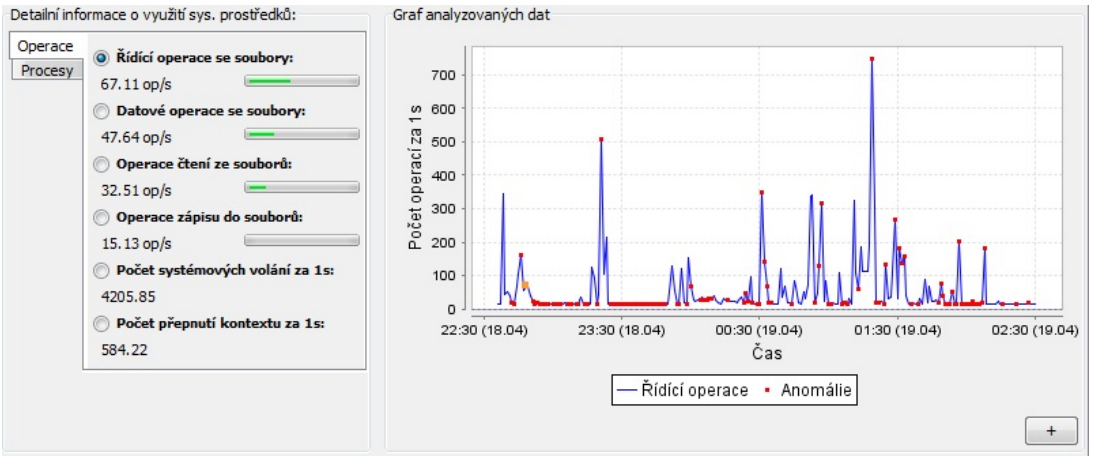
Spodní polovina střední části okna serverové části aplikace je určena pro zobrazování detailních informací o vybraném záznamu, zvoleném ve vrchní textové tabulce a grafické reprezentaci naměřených hodnot a výsledků analýzy anomálií ve formě grafů. Spolu s panely se zde vyskytuje série přepínačů, na základě jejichž přepnutí je zobrazován obsah. Komponenta pro realizaci grafů je ovladatelná pomocí myši, umožňuje základní operace jako zvětšování, zmenšování či posun úseku grafu a za použití pravého tlačítka myši pak operace jako uložení grafu do souboru, tisk nebo zkopírování do schránky.

Přepnutí prvního přepínače nazvaného „Grafy analýzy záznamů“ vede k zobrazení celkového výsledku analýzy dat jednotlivými detektory anomálií. Je zobrazen graf s vynesenými vzdálenostmi od prahu normálního chování pro všechny analyzované hodnoty daného klienta. Tento graf má největší význam, neboť nabízí přehledným způsobem celkový pohled na detekované anomálie v jednotlivých okamžicích. Pomocí ovládacích prvků je možno vybrat detektory, které budou na grafu vykresleny.



Obrázek 6.17: Zobrazení grafu celkového výsledku analýzy

Přepínačem „Grafy vstupních dat“ jsou přístupné původní naměřené hodnoty pro zvolený záznam jak v grafické, tak i textové formě. Použitím přepínačů ve spodní části je umožněna navigace mezi jednotlivými kategoriemi měřených dat. Pomocí červených bodů jsou vyznačeny okamžiky, ve kterých byly záznamy vyhodnoceny jako abnormální.



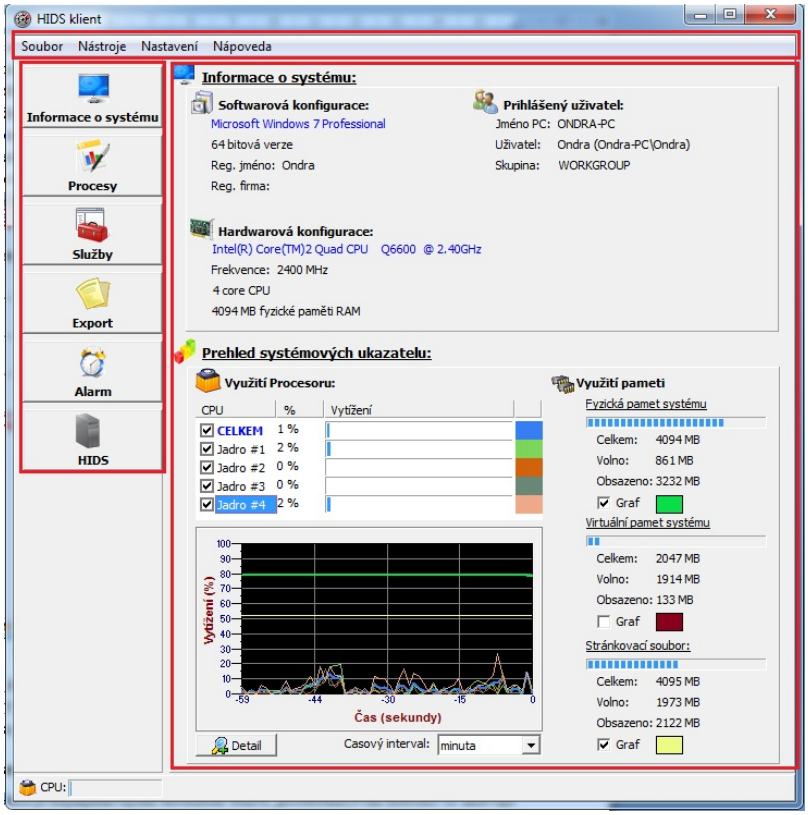
Obrázek 6.18: Zobrazení grafu vstupních hodnot pro vybraný parametr

6.7.2 Grafické uživatelské rozhraní klientské části

Klientská aplikace nabízí jednoduché grafické rozhraní, sloužící především k přehlednému zobrazení monitorovaných aktivit na klientské stanici. Aplikace je rozvržena do několika modulů podle oboru sledovaných dat. Na tomto místě bude pouze ve zkratce popsáno základní uživatelské rozhraní klientské aplikace a modul související s HIDS funkcionalitou.

6.7.2.1 Hlavní okno aplikace

Hlavní okno aplikace je v horní části tvořeno systémovou nabídkou, s jejíž pomocí je možno s aplikací pracovat a nastavovat ji. Nabídka obsahuje položky “soubor“, “nástroje“, “nastavení“ a “nápověda“. Pod položkou “soubor“ se skrývají položky pro zálohování dat a ukončení aplikace. Zálohování dat umožňuje uložení datových souborů se statistickými daty na zvolené médium. Položka “nástroje“ obsahuje položky, jejichž aktivací se aplikace přepne na zvolený modul. Tato nabídka je ekvivalentem k postrannímu menu aplikace, jež bude zmíněno níže. Položka “nastavení“ aktivuje dialog, s jehož pomocí je možno nastavit chování aplikace [16].



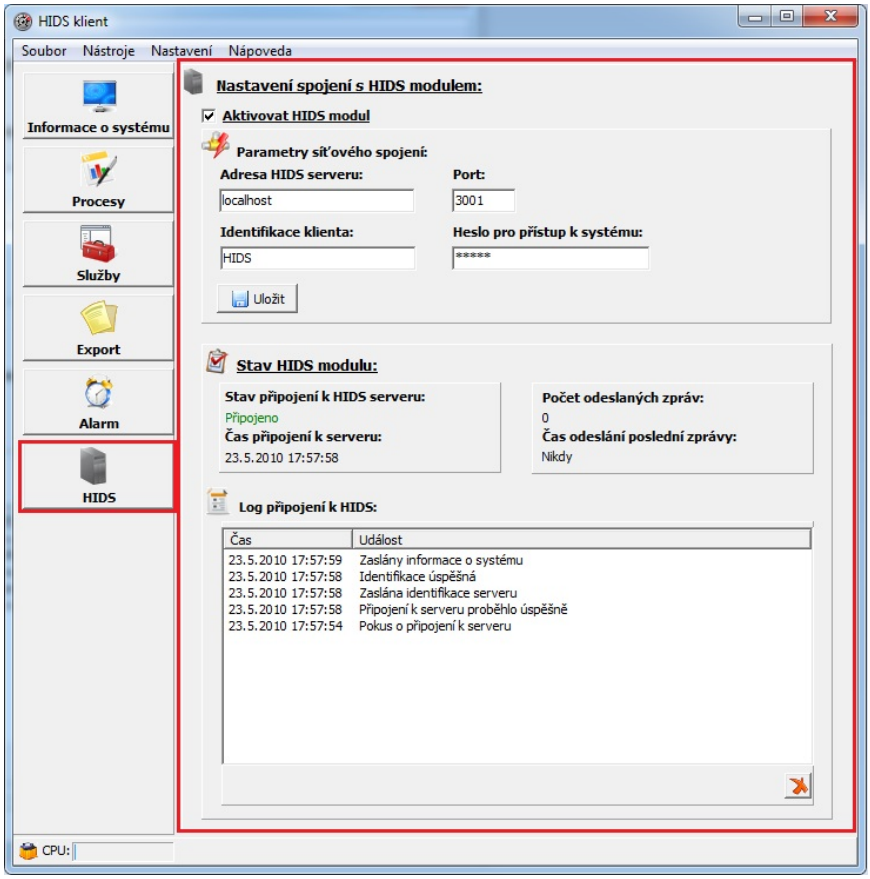
Obrázek 6.19: Hlavní okno klientské části aplikace

Levá část okna je tvořena pěti tlačítky menu, sloužícími k navigaci po aplikaci. Volbou tlačítka je aktivován příslušný modul a uživateli jsou prezentovány požadované informace.

Pravá část okna je určena pro zobrazení panelů jednotlivých modulů

6.7.2.2 Modul nastavení HIDS

Z hlediska HIDS aplikace je nejzajímavějším modulem HIDS, prostřednictvím kterého se aktivuje zasílání naměřených dat serverové části HIDS aplikace. V modulu lze specifikovat adresu serveru a port, na kterém jsou přijímána data, dále se zde nastavuje jméno, jež identifikuje klientskou stanici a heslo, které umožňuje připojení k serveru. Po nastavení potřebných údajů jsou zobrazeny informace o stavu, v němž se nachází spojení mezi klientskou a serverovou částí.



Obrázek 6.20: Nastavení komunikace se serverovou částí HIDS aplikace

6.8 Testování aplikace a dosažené výsledky

Výsledná aplikace byla dlouhodobě testována z různých pohledů, kdy byly ověřovány její vlastnosti z odlišné perspektivy.

První typ testování představoval ověření celkové funkčnosti aplikace a uživatelského rozhraní. Pro tento druh testů bylo používáno rozdílných hardwarových i softwarových konfigurací testovacích systémů a byl kladen důraz na ověření kompatibility výsledného produktu. Při testování bylo používáno dobrovolníků, kteří měli možnost s aplikací a jejím ovládáním experimentovat a na základě jejich podnětů došlo k drobným úpravám grafického uživatelského rozhraní aplikace i některých jejich funkcí.

Druhým typem testů byly ověřovány detekční schopnosti aplikace a celková přesnost detekce. Aplikace byla testována při různých režimech provozu systému. Tomuto typu testování se podrobněji věnuje následující část kapitoly, kde jsou detekční schopnosti demonstrovány na několika modelových testech, které si berou za cíl přiblížit se reálným situacím, jež mohou nastat.

Třetím a posledním typem testování, které bylo prováděno, je testování výkonnosti aplikace. Aplikace byla testována s různým počtem připojených klientů při provádění různých operací detektoru. Těmto testům se věnuje poslední část této kapitoly.

6.8.1 Testy detekčních schopností HIDS aplikace

Tato část kapitoly si bere za cíl zhodnotit reálné detekční schopnosti implementované HIDS aplikace. Pro účely testování byl vytvořen model chování, který odpovídá běžnému uživateli, využívajícímu základních kancelářských a multimediálních aplikací. Uživatel tohoto profilu běžně prohlíží obsah

internetových stránek za použití prohlížeče Mozilla Firefox, při práci ve svém profilu přehrává hudbu a sleduje filmy v běžném rozlišení. Mezi jeho standardní aktivity patří zpracovávání textu pomocí textového procesoru Open Office a prohlížení dokumentů ve formátu pdf. Tato data jsou uložena na souborovém serveru a uživatel k nim přistupuje pomocí lokální sítě. V profilu jsou zaznamenány jak období, kdy je uživatel aktivní, tak i období, kdy nikdo s počítačem nepracuje. Na základě těchto činností byl vytvořen profil uživatelského chování pomocí demonstrační HIDS aplikace.

Za trénovací množinu byl zvolen soubor 1076 záznamů normálního chování uživatele. Pro trénování byly nastaveny parametry detekčního jádra systému na následující hodnoty:

1. fáze učení		2. fáze učení	
Parametr	**Hodnota**	**Parametr**	**Hodnota**
Počet iterací	400	**Počet iterací**	1000
Počáteční poloměr	10,0	**Počáteční poloměr**	5,0
Konečný poloměr	5,0	**Konečný poloměr**	4,85
Počáteční faktor učení	0,5	**Počáteční faktor učení**	0,5
Konečný faktor učení	0,1	**Konečný faktor učení**	0,1

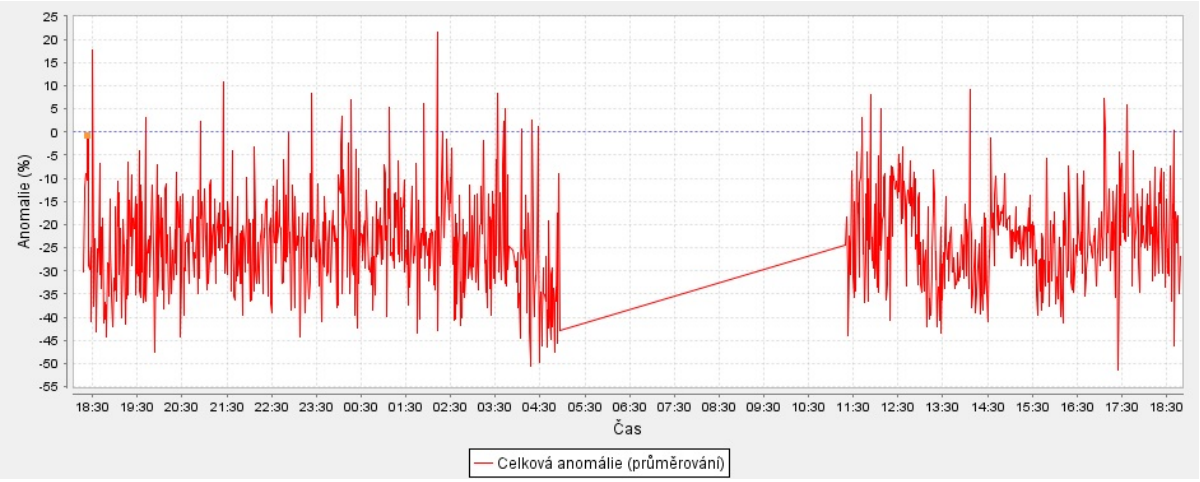
6.9: Nastavení detektoru anomálií

Kvalitu získaného modelu pak mohou charakterizovat počty záznamů, jejichž vzdálenost od středu clusteru normálního chování nepřekračuje prahovou hodnotu. Použitý model pak při velikosti korekčního faktoru 1.0 dosahoval následujících hodnot:

Detektor	Vyhovující záznamy	Nevyhovující záznamy	Přesnost modelu
Procesor	902	174	83,82 %
Pevný disk	933	143	86,71 %
Síť	959	117	89,12 %
Systémové prostředky	883	193	82,06 %
Přístupy k systému	1055	21	98,04 %
Log	1072	4	99,62 %
Celkový	961	115	89,93 %

Tabulka 6.10: Statistické hodnoty naučeného modelu chování

Přesnost modelů jednotlivých detektorů je dána kromě nastavení parametrů učení také rozmanitostí dat, která zpracovávají. Pomocí korekčního faktoru je možno posunout prahovou hodnotu a tím zvýšit počet vstupních hodnot, které vyhoví kritériím. Zvýšením prahu ale pak může docházet k současnému zvýšení počtu záznamů, které jsou falešně negativní.



Obrázek 6.21: Graf vzdáleností vstupních hodnot naučeného modelu od prahu normálního chování

Na získaném modelu chování byla provedena série čtyř rozdílných testů, kdy v následující části kapitoly je vždy ke každému testu prezentován graf, popis aktivit které probíhaly a vyhodnocení dosažených výsledků.

Při jednotlivých testech byly sledovány počty falešně pozitivních a falešně negativních vzorků a na jejich základě vypočteny ukazatele efektivity „Detection rate“ a „False alarm rate“ [28].

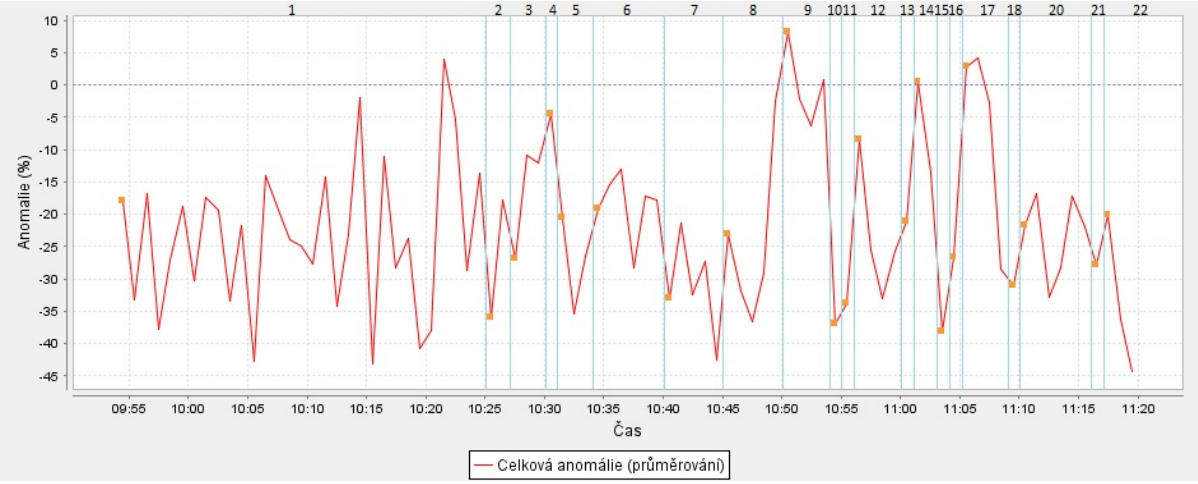
$$Detection\ rate = \frac{TP}{TP + FN} \tag{6.16}$$

$$False\ alarm\ rate = \frac{FP}{TN + FP} \tag{6.17}$$

kde TN značí počet pravdivě negativních vzorků, TP počet pravdivě pozitivních vzorků, FN počet falešně negativních a FP počet falešně pozitivních vzorků.

6.8.1.1 Test dat neobsahujících žádnou abnormální aktivitu

První test se zaměřuje na chování, jež odpovídá standardnímu chování uživatele, a proto by mělo být označeno detekčním systémem jako normální. V průběhu testu docházelo k běžným aktivitám, jakými jsou sledování videa či prohlížení internetových stránek.



Obrázek 6.22: Graf vzdáleností vstupních hodnot od prahu (průměrování)

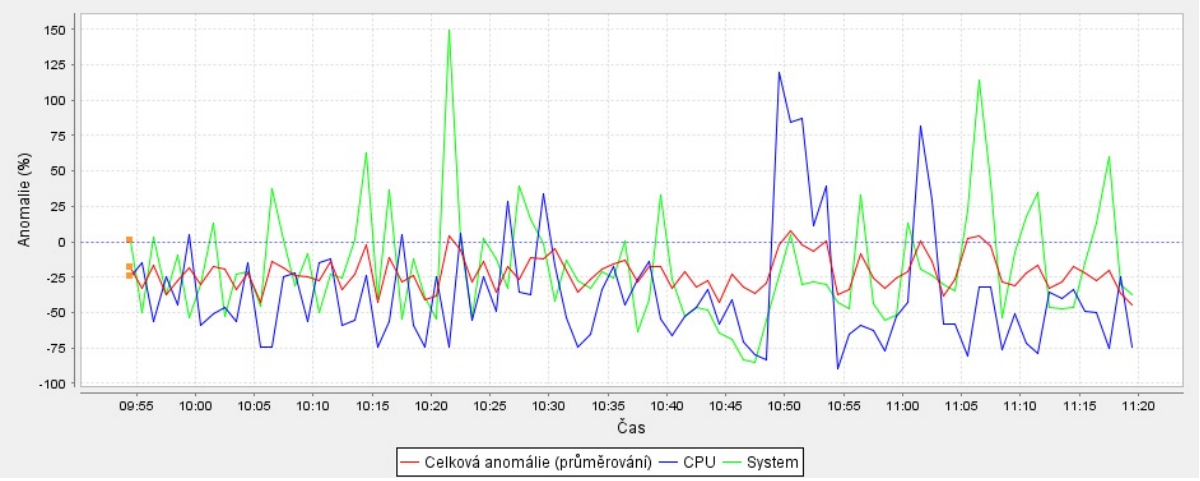
Kompletní výčet aktivit, které při testu probíhaly, je následující, přičemž jako konečný výsledek detekce je brána celková hodnota anomality, získaná výpočtem průměru hodnot jednotlivých detektorů:

č.	Čas		Popis	Skutečná anomálie	Detekovaná anomálie	Počet detekovaných nesprávně
	start	konec				
1	9:55	10:25	Žádná aktivita	NE	ČÁSTEČNĚ	1 (do +5% od T)
2	10:26	10:27	Spuštění Open Office	NE	NE	0
3	10:28	10:30	Načtení dokumentu ze sítě a editace	NE	NE	0
4	10:31	10:31	Ukončení Open Office	NE	NE	0
5	10:32	10:34	Spuštění Mozilla Firefox a prohlížení internetových stránek	NE	NE	0
6	10:35	10:40	Žádná aktivita	NE	NE	0
7	10:41	10:45	Přehrávání videa v avi formátu	NE	NE	0
8	10:46	10:50	Prohlížení internetových stránek	NE	NE	0

9	10:51	10:54	Přehrávání mp3 ve Windows media playeru + prohlížení internetových stránek	NE	ČÁSTEČNĚ	2 (do +10% od T)
10	10:55	10:55	Ukončení Windows media playeru	NE	NE	0
11	10:56	10:56	Otevření pdf souboru v Acrobat Reader	NE	NE	0
12	10:57	11:00	Prohlížení pdf souboru a prohlížení internetových stránek	NE	NE	0
13	11:01	11:01	Ukončení Mozilla Firefox a Acrobat Reader	NE	NE	0
14	11:02	11:03	Procházení sdílených síťových složek	NE	ČÁSTEČNĚ	1 (do 5% od T)
15	11:04	11:04	Kopírování 3 dokumentů ze sítě na disk	NE	NE	0
16	11:05	11:05	Spuštění Open Office	NE	NE	0
17	11:06	11:09	Editace zkopírovaných dokumentů v Open Office	NE	ČÁSTEČNĚ	2 (do 5% od T)
18	11:10	11:10	Uložení dokumentů a ukončení Open Office	NE	NE	0
19	11:11	11:16	Přehrávání videa v avi formátu	NE	NE	0
20	11:17	11:17	Vymazání 3 dokumentů	NE	NE	0
21	11:18	11:20	Žádná aktivita	NE	NE	0

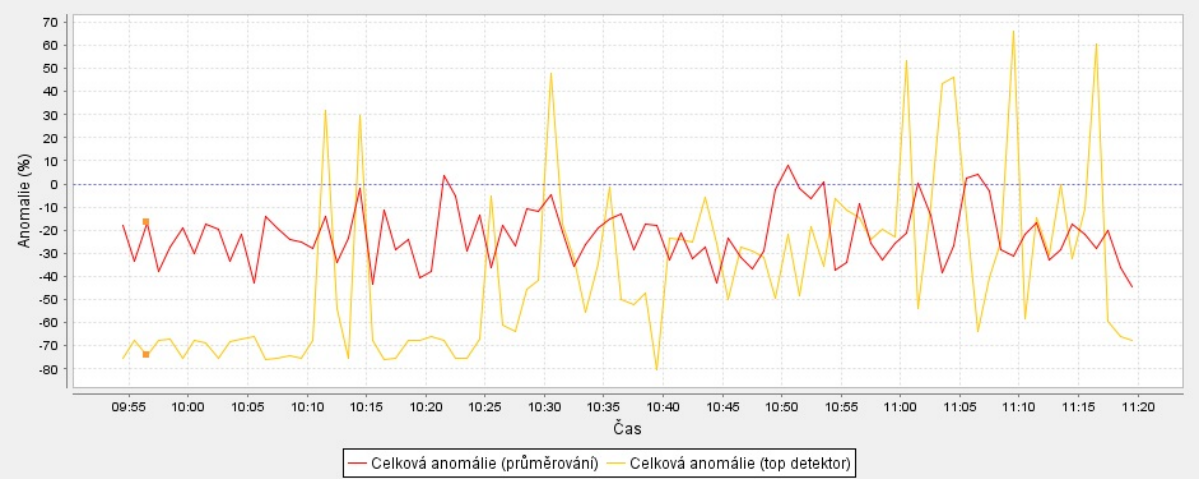
Tabulka 6.11: Časový průběh aktivit během prvního testu detektoru

Z tabulky 6.11 plyne, že naprostá většina záznamů byla klasifikována jako normální chování uživatel. V analyzovaných datech se vyskytlo několik záznamů, které velmi mírně převyšovaly prahovou hodnotu. Jednalo se především o operace, při kterých docházelo ke spouštění aplikací nebo načítání dat. Anomálie byly v těchto datech zaznamenány zejména v částech detektoru, analyzujících informace o procesoru a systémových prostředcích. Vzdálenosti od prahu normálního chování jsou zobrazeny na grafu 6.23.



Obrázek 6.23: Graf vzdáleností vstupních hodnot od prahu detektorů nižší úrovně

Na následujícím grafu je pro úplnost provedeno porovnání metody průměrování a alternativní implementované metody, založené na přidání detektoru vyšší úrovně. Bohužel výsledky při použití metody detektoru vyšší úrovně jsou nejednoznačné a vyskytuje se větší množství falešně negativních výsledků detekce.



Obrázek 6.24: Graf vzdáleností vstupních hodnot od prahu (porovnání detekčních metod)

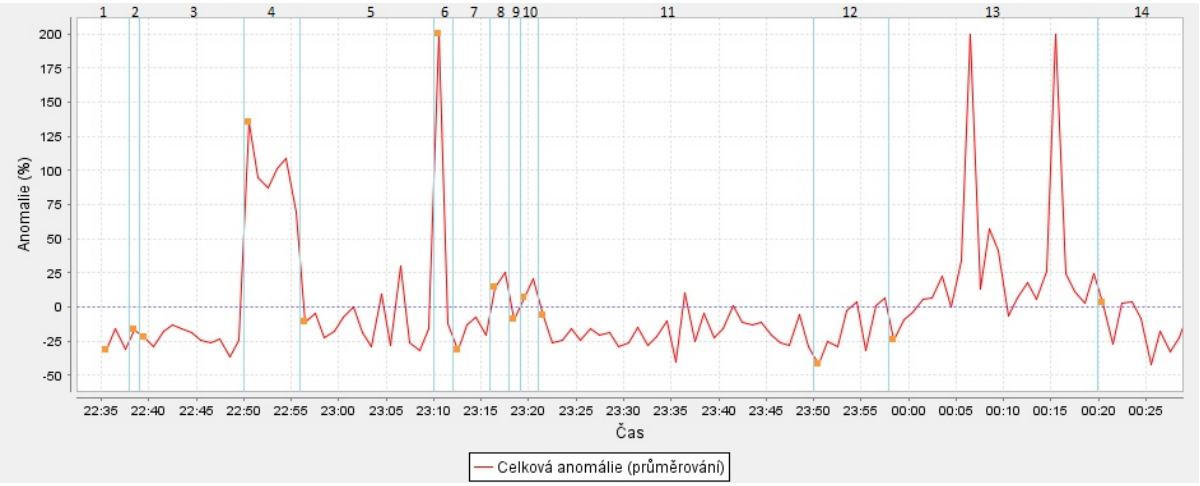
V tabulce 6.12 jsou vyčísleny počty falešně pozitivních výsledků detekce. Dále je pro úplnost u obou metod vypočtena hodnota False alarm rate a Detection rate.

	Počet hodnot	Falešně pozitivní	Falešně negativní	Pravdivě pozitivní	Pravdivě negativní	False alarm rate	Detection rate
Průměr hodnot	86	6	0	0	80	0,069	-
Detektor vyšší úrovně	86	8	0	0	78	0,093	-

Tabulka 6.12: Vyhodnocení přesnosti detekce

6.8.1.2 Test dat obsahujících podezřelé uživatelské chování

Druhý ze série testů se zaměřuje na podezřelé chování uživatele. V rámci testu byly prováděny takové aktivity, které se vymykají standardnímu chování. Snahou bylo nasimulovat aktivity, které indikují pokus o krádež dat, zásahy do nastavení operačního systému a pokusy o neautorizované využití zdrojů. Pro lepší přehlednost byl test rozdělen do dvou na sebe navazujících částí.



Obrázek 6.25: Graf vzdáleností vstupních hodnot od prahu (průměrování) - 1. část

Kompletní výčet aktivit, které v průběhu první části tohoto testu probíhaly je následující:

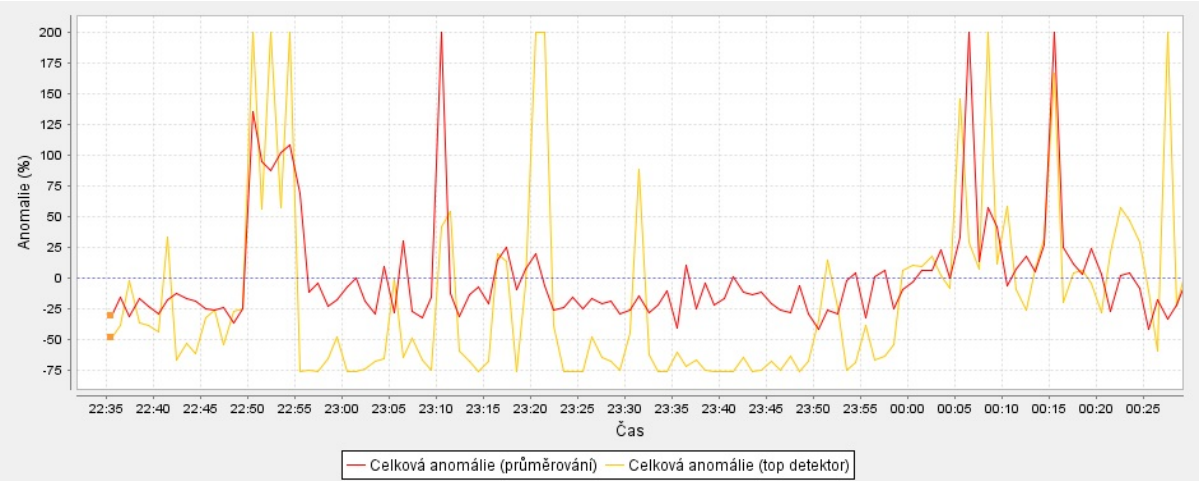
č.	Čas		Popis	Skutečná anomálie	Detekovaná anomálie	Počet detekovaných nesprávně
	start	konec				
1	22:36	22:38	Žádná aktivita	NE	NE	0
2	22:39	22:39	Spuštění Mozilla Firefox a načtení uložených stránek	NE	NE	0

3	22:40	22:50	Prohlížení internetových stránek	NE	NE	0
4	22:51	22:56	Kopírování 2GB dat ze sítě	ANO	ANO	0
5	22:57	23:10	Žádná aktivita	NE	ČÁSTEČNĚ	2
6	23:11	23:12	Vymazání 10 souborů	ANO	ČÁSTEČNĚ	1
7	23:13	23:16	Žádná aktivita	NE	NE	0
8	23:17	23:18	Čištění registru a souborového systému	ANO	ANO	0
9	23:19	23:19	Žádná aktivita	NE	NE	0
10	23:20	23:21	Spuštěno vyhledávání souboru obsahujícího textový řetězec	ANO	ANO	0
11	23:22	23:50	Žádná aktivita	NE	ČÁSTEČNĚ	2 (do +10% od T)
12	23:51	23:58	Prohlížení internetových stránek	NE	ČÁSTEČNÉ	2 (do +5% od T)
13	23:59	0:20	Výměna dat přes P2P - torrent	ANO	ČÁSTEČNĚ	5
14	0:21	0:29	Žádná aktivita	NE	ČÁSTEČNĚ	2 (do +5% od T)

Tabulka 6.13: Časový průběh aktivit během druhého testu detektoru – 1. část

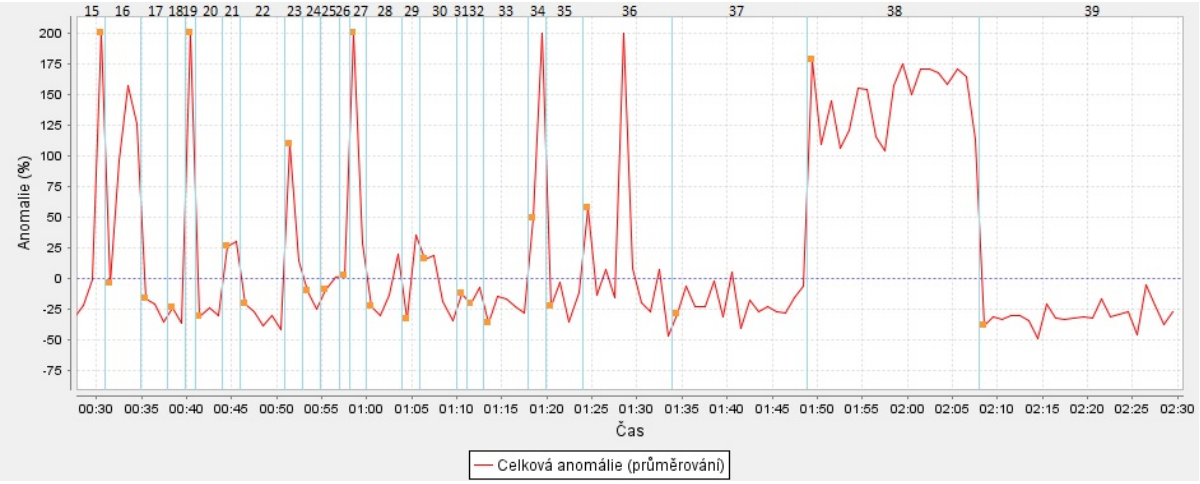
Z tabulky 6.13 je patrné, že všechny situace vyskytujícího se anomální chování byly detektorem rozpoznány. V případech některých aktivit, jako je například výměna dat pomocí torrentu, nebyly detekovány anomálie po celou dobu běhu aplikace, přičemž pravděpodobnou příčinnou je rozdílná aktivita klienta torrentu v jednotlivých časových okamžicích. Byly zaznamenány falešně pozitivní výsledky detekce v některých částech testu, jež měly dle předpokladu vykazovat normální chování. Tato měření se ve valné většině vyznačují vzdáleností převyšující práh maximálně o 10 procent, z čehož lze usuzovat, že mohla být prováděna nezjištěná aktivita operačního systému na pozadí, případně lze uvažovat o nepřesnosti detekční metody.

Pro srovnání je zde uváděn graf obsahující výsledky detekce pomocí detektoru vyšší úrovně. V této části testu dosáhnul velmi dobrých výsledků, kdy úspěšně odhalil ať již kompletně nebo částečně časové úseky, obsahující anomální hodnoty.



Obrázek 6.26: Graf vzdáleností vstupních hodnot od prahu (porovnání detekčních metod) – 1. část

Průběh aktivit vykonávaných v druhé části tohoto testu lze graficky reprezentovat grafem 6.27.



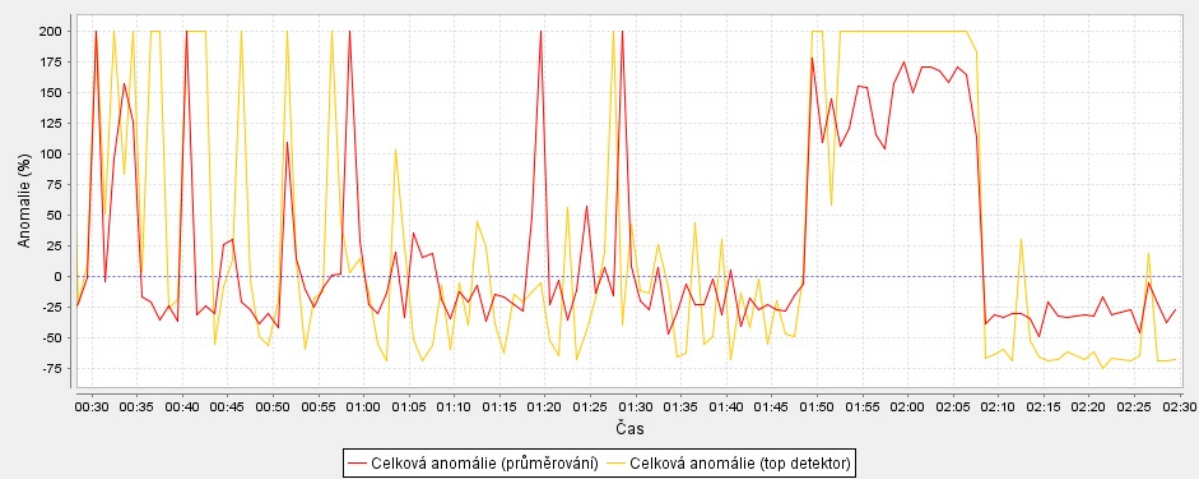
Obrázek 6.27: Graf vzdáleností vstupních hodnot od prahu (průměrování) – 2. část

Kompletní výčet aktivit druhé části testu zachycených grafem a jejich popis je uveden v následující tabulce:

č.	Čas		Popis	Skutečná anomálie	Detekovaná anomálie	Počet detekovaných nesprávně
	start	Konec				
15	0:31	0:31	Připojení externího pevného disku	ANO	ANO	0
16	0:32	0:35	Kopírování dat z externího disku	ANO	ČÁSTEČNĚ	1 (do -5% od T)
17	0:36	0:38	Žádná aktivita	NE	NE	0
18	0:39	0:40	Prohlížení internetových stránek	NE	NE	0
19	0:41	0:41	Odebrání externího disku	ANO	ANO	0
20	0:42	0:44	Žádná aktivita	NE	NE	0
21	0:45	0:46	Zásah do souborů v adresáři Windows	ANO	ANO	0
22	0:47	0:51	Prohlížení internetových stránek	NE	NE	0
23	0:52	0:53	Čištění registru a souborového systému	ANO	ANO	0
24	0:54	0:55	Žádná aktivita	NE	NE	0
25	0:56	0:57	Přidání nové naplánované úlohy	ANO	ČÁSTEČNĚ	1 (do -10% od T)
26	0:58	0:58	Žádná aktivita	NE	ANO	1 (do +5% od T)
27	0:59	1:00	Změna konfigurace aktualizací OS	ANO	ANO	0
28	1:01	1:04	Žádná aktivita	NE	ČÁSTEČNĚ	1
29	1:05	1:06	Manipulace s nastavením HW – zákáz zvukové karty	ANO	ČÁSTEČNĚ	1
30	1:07	1:10	Žádná aktivita	NE	ČÁSTEČNĚ	2
31	1:11	1:11	Spuštění Open Office	NE	NE	0
32	1:12	1:13	Načtení dokumentu ze sítě a editace	NE	NE	0
33	1:14	1:18	Přehrávání videa v avi formátu	NE	NE	0
34	1:19	1:20	Úprava nastavení sdílených složek	ANO	ANO	0
35	1:21	1:24	Žádná aktivita	NE	NE	0
36	1:25	1:34	Výměna dat přes P2P – torrent	ANO	ČÁSTEČNĚ	5
37	1:35	1:49	Žádná aktivita	NE	ČÁSTEČNĚ	1 (do +10% od T)
38	1:50	2:08	Kopírování velkého množství dat na síťový server	ANO	ANO	0
39	2:09	2:30	Žádná aktivita	NE	NE	0

Tabulka 6.14: Časový průběh aktivit během druhého testu detektoru – 2. část

Stejně jako v případě první části tohoto testu byly ať již částečně nebo úplně detekovány všechny předpokládané anomálie. Nejproblematičtější byla detekce P2P torrentu, kde není výsledek detekce zcela jednoznačný. Ve vzorcích se opět vyskytuje několik falešně pozitivních i falešně negativních záznamů, kdy ve většině případů není vzdálenost od stanoveného prahu větší než 5 až 10 procent.



Obrázek 6.28: Graf vzdáleností vstupních hodnot od prahu (porovnání detekčních metod) – 2. část

Při srovnání metody průměrování a detektoru vyšší úrovně je z grafu patrné, že metoda detektoru vyšší úrovně nebyla schopna detekovat některé aktivity, jejichž příkladem může být manipulace s nastavením sdílených složek, výsledky detekce také obsahují více falešně pozitivních záznamů.

Největší vliv na detekované anomálie měly detektory zpracovávající data z procesoru, pevných disků a informace o systémových prostředcích i přístupech k systému.

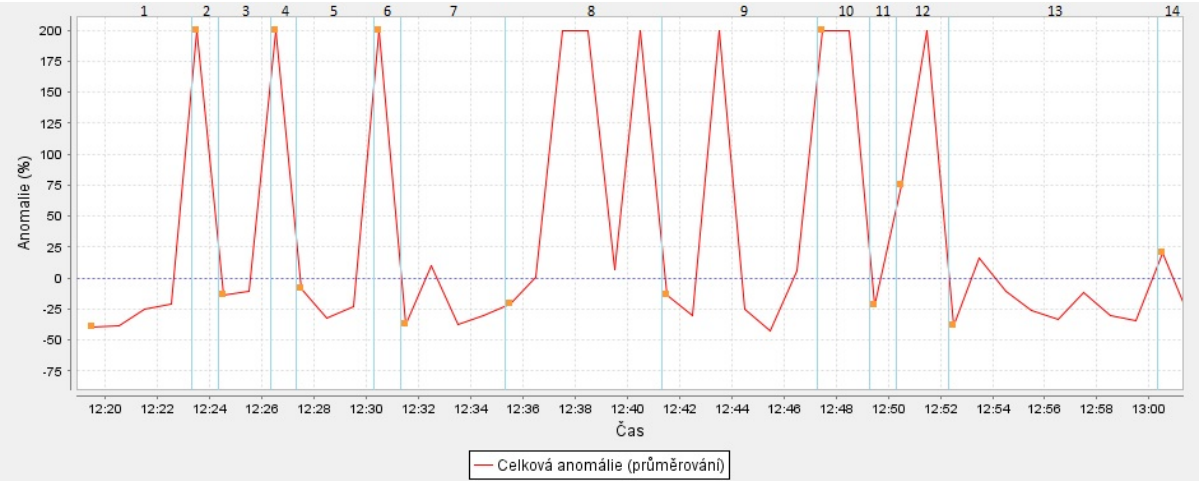
V následující tabulce jsou vyčísleny počty falešně pozitivních výsledků detekce. Dále je pro úplnost u obou metod vypočtena hodnota False alarm rate a Detection rate.

	Počet hodnot	Falešně pozitivní	Falešně negativní	Pravdivě pozitivní	Pravdivě negativní	False alarm rate	Detection rate
Průměr hodnot	234	16	13	68	137	0,104	0,839
Detektor vyšší úrovně	234	24	19	67	124	0,162	0,779

Tabulka 6.15: Vyhodnocení přesnosti detekce

6.8.1.3 Test dat obsahujících podezřelé chování aplikací

Třetí test se zabývá schopnostmi detekční HIDS aplikace rozpoznat anomální chování aplikací a systému. Snahou bylo nasimulovat situace, které mohou nastat při nekorektním chování aplikací a mohou způsobit jejich pád nebo narušení běhu celého operačního systému. V rámci tohoto testu bylo také sledováno, zda HIDS aplikace zaznamená snahy aplikací o nežádoucí činnost, jako je modifikace nastavení operačního systému nebo instalace nežádoucích komponent. Po testování pádu aplikací a systému bylo použito zásahu do jimi alokovaných oblastí paměti. Pro lepší přehlednost byl časový úsek testu opět rozdělen do dvou na sebe navazujících grafů.



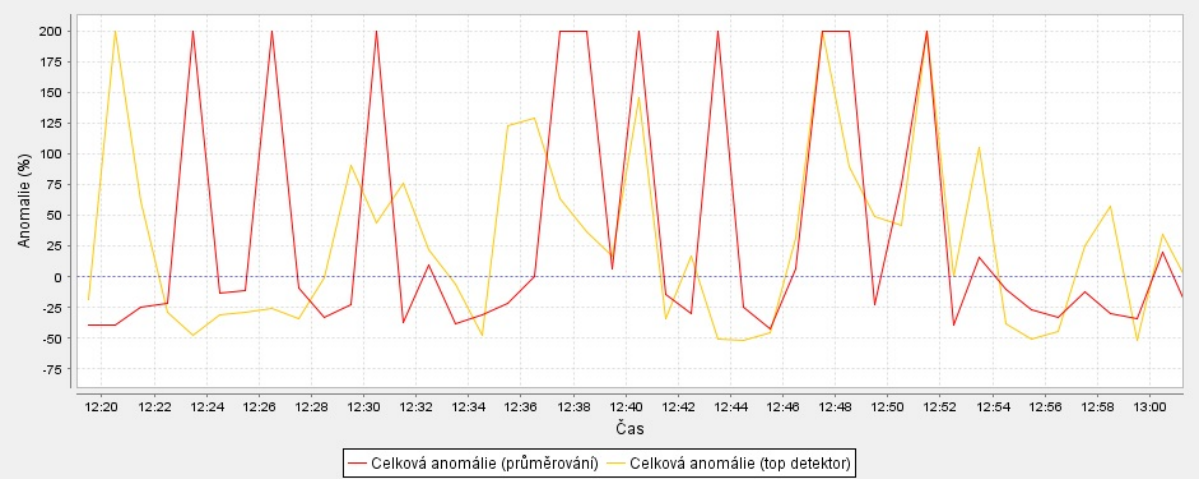
Obrázek 6.29: Graf vzdáleností vstupních hodnot od prahu (průměrování) – 1. část

Kompletní výčet a popis aktivit zachycených grafem během první části testu je uveden v následující tabulce:

č.	Čas		Popis	Skutečná anomálie	Detekovaná anomálie	Počet detekovaných nesprávně
	start	konec				
1	12:20	12:23	Žádná aktivita	NE	NE	0
2	12:24	12:24	Pád aplikace Mozilla Firefox	ANO	ANO	0
3	12:25	12:26	Žádná aktivita	NE	NE	0
4	12:27	12:27	Nežádoucí instalace Ask toolbaru	ANO	ANO	0
5	12:28	12:30	Žádná aktivita	NE	NE	0
6	12:31	12:31	Pád aplikace přistupující mimo rozsah pole	ANO	ANO	0
7	12:32	12:35	Žádná aktivita	NE	ČÁSTEČNĚ	1 (do +10% od T)
8	12:36	12:41	Pád aplikace obsahující memory leak	ANO	ČÁSTEČNĚ	1
9	12:42	12:47	Přehrávání videa v avi formátu	NE	ČÁSTEČNĚ	2
10	12:48	12:49	Neúspěšný pokus o odinstalování Ask toolbaru	ANO	ANO	0
11	12:50	12:50	Žádná aktivita	NE	NE	0
12	12:51	12:52	Úspěšný pokus o odinstalování Ask toolbaru	ANO	ANO	0
13	12:53	13:00	Prohlížení internetových stránek	NE	ČÁSTEČNĚ	1
14	13:01	13:01	Pád aplikace Mozilla Firefox a její restartování	ANO	ANO	0

Tabulka 6.16: Časový průběh aktivit během třetího testu detektoru – 1. část

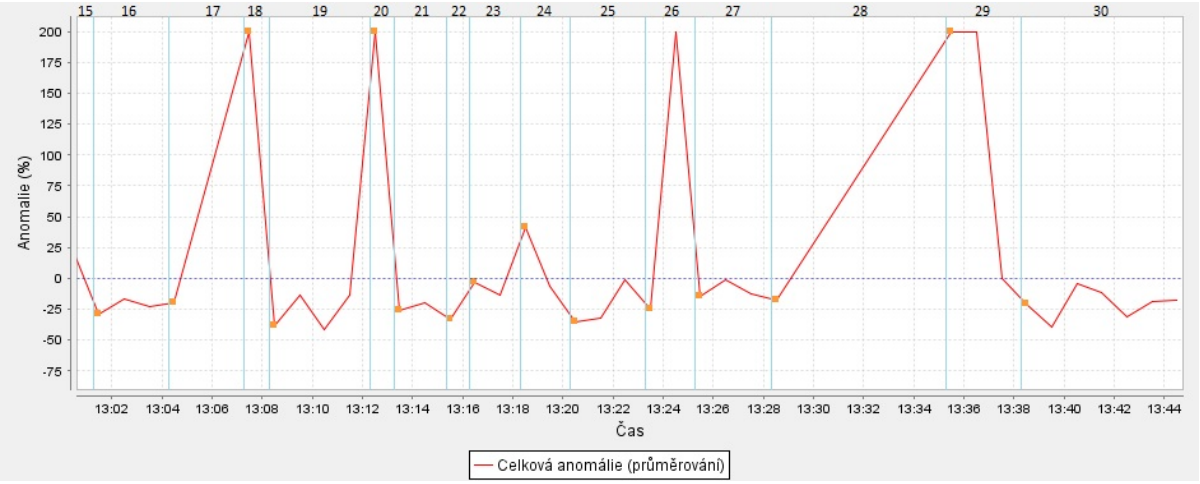
Všechny předpokládané anomálie byly HIDS aplikací ať již částečně nebo úplně detekovány. Detekce vykazovala v případě několika měření falešně pozitivní výsledek, kdy nejvyšší odchylka od prahu byla zaznamenána v 9. části testu, kdy při přehrávání videa došlo zřejmě k načítání dat, odložených při předchozí části testu do stránkovacího souboru. Část testu číslo 8 totiž sledovala aplikaci, která obsahovala memory leak a alokovala velké množství operační paměti.



Obrázek 6.30: Graf vzdáleností vstupních hodnot od prahu (porovnání detekčních metod) – 1. část

Z grafu srovnání metody průměrování vůči metodě používající detektor vyšší úrovně jsou viditelné slabší výsledky detektoru vyšší úrovně v této části testu. V první části nebyl detektor vyšší úrovně schopen zachytit některá anomální měření a naopak některá vyhodnotil jako falešně pozitivní, ve druhé části jsou pak jeho výsledky již na lepší úrovni.

Průběh aktivit vykonávaných v druhé části tohoto testu lze graficky vyjádřit následujícím grafem:



Obrázek 6.31: Graf vzdáleností vstupních hodnot od prahu (průměrování) – 2. Část

Kompletní výčet aktivit druhé části testu zachycených grafem a jejich popis je uveden v následující tabulce:

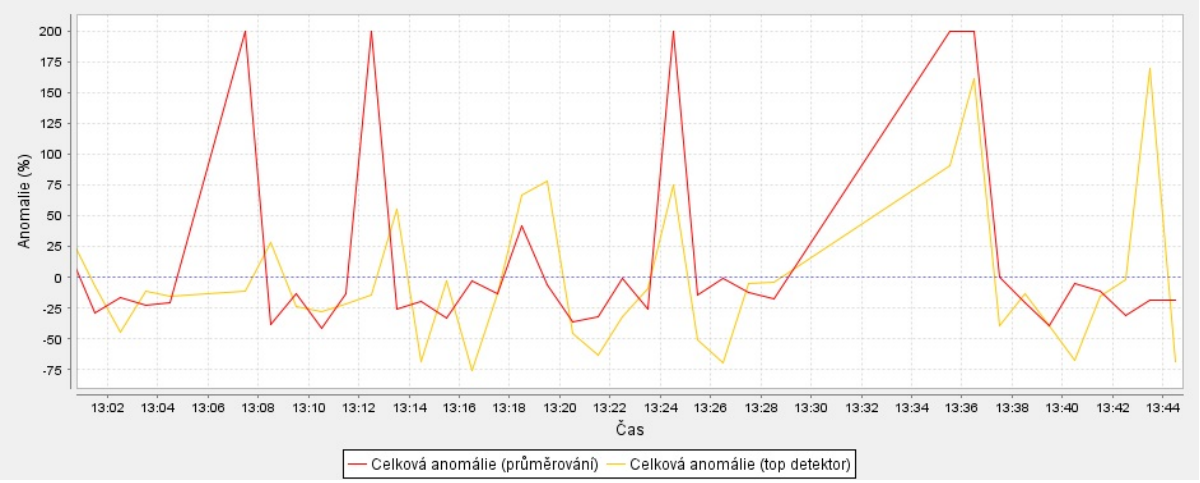
č.	Čas		Popis	Skutečná anomálie	Detekovaná anomálie	Počet detekovaných nesprávně
	start	konec				
15	13:01	13:01	Pád aplikace Mozilla Firefox a její restartování	ANO	ANO	0
16	13:02	13:04	Žádná aktivita	NE	NE	0
17	13:05	13:07	Pád operačního systému Windows a restart, (13:06–13:07) nedostupná data	ANO	NE	1
18	13:08	13:08	Start operačního systému a načtení událostí z EventLogu	ANO	ANO	0
19	13:09	13:12	Žádná aktivita	NE	NE	0
20	13:13	13:13	Modifikace nastavení firewallu	ANO	ANO	0
21	13:14	13:15	Žádná aktivita	NE	NE	0

22	13:16	13:16	Modifikace nastavení uživatelských účtů	ANO	NE	1
23	13:17	13:18	Žádná aktivita	NE	NE	0
24	13:19	13:20	Modifikace registru operačního systému	ANO	ČÁSTEČNĚ	1 (do -10% od T)
25	13:21	13:23	Prohlížení internetových stránek	NE	NE	0
26	13:24	13:25	Instalace nežádoucí aplikace a registrace jejího spuštění po startu systému	ANO	ČÁSTEČNĚ	1
27	13:26	13:28	Žádná aktivita	NE	NE	0
28	13:29	13:35	Pád operačního systému, zobrazení BOSD obrazovky a následný restart, (13:30–13:35) nedostupná data	ANO	NE	1
29	13:36	13:38	Start operačního systému a načtení událostí z EventLogu	ANO	ANO	0
30	13:39	13:45	Žádná aktivita	NE	ANO	0

Tabulka 6.17: Časový průběh aktivit během třetího testu detektoru – 2. část

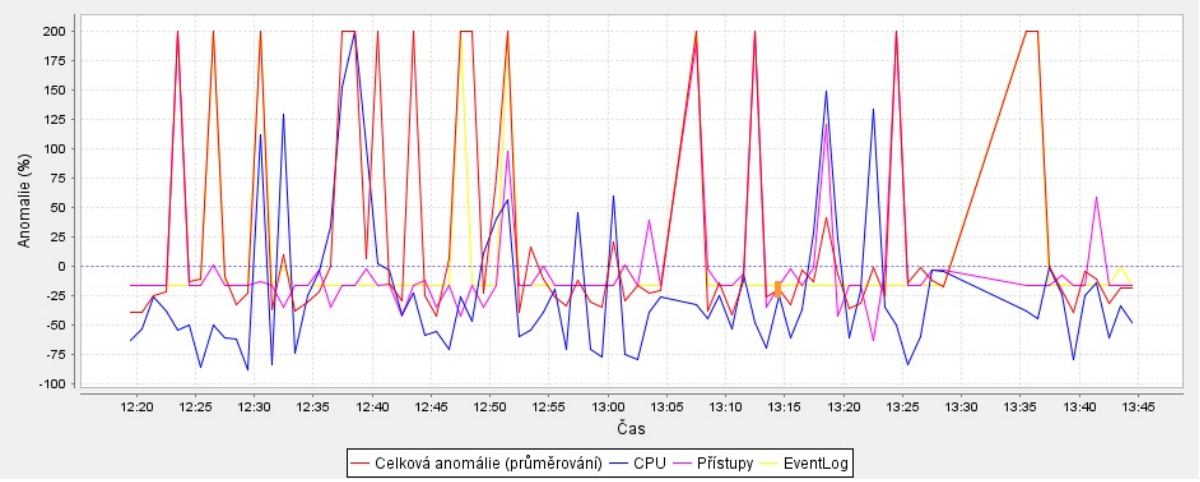
Během druhé části tohoto testu nebyly detektorem detekovány některé záznamy jako anomální. Detektor využívající metodu průměrování nedetekoval změnu nastavení uživatelských účtů a nebylo možno detekovat okamžik, ve kterém došlo k pádu operačního systému. Okamžik pádu systému nebyl detekován z důvodu okamžitého restartu počítače, ovšem první okamžiky po startu systému byly označeny jako anomální, neboť detekční systém vyhodnotil záznamy EventLogu jako náležející anomálnímu chování a tak je pád systému v grafu zřetelně viditelný.

Pro úplnost je zde uvedeno srovnání implementovaných detekčních metod. Detektor vyšší úrovně detekoval anomálie po pádu systému a firewallu s jistým zpožděním a v konečné části označil některé záznamy jako falešně pozitivní. V této části testu se jeho detekční schopnosti jeví jako srovnatelné s metodou průměrování a na grafu jsou anomálie zřetelně viditelné na obrázku 6.32.



Obrázek 6.32: Graf vzdáleností vstupních hodnot od prahu (porovnání detekčních metod) – 2. část

Většina anomálních záznamů v obou částech testu byla detekována na základě dat analyzovaných pomocí detektorů zpracovávajících data o využití procesoru, o přístupech k systémovým oblastem a záznamech v logu operačního systému. Graf vzdáleností vstupních hodnot od prahu normálního chování pro jednotlivé detektory je zobrazen na obrázku 6.33.



Obrázek 6.33: Graf vzdáleností vstupních hodnot od prahu detektorů nižší úrovně

V následující tabulce jsou vyčísleny počty falešně pozitivních výsledků detekce. Dále je pro úplnost u obou metod vypočtena hodnota False alarm rate a Detection rate.

	Počet hodnot	Falešně pozitivní	Falešně negativní	Pravdivě pozitivní	Pravdivě negativní	False alarm rate	Detection rate
Průměr hodnot	78	4	6	20	48	0,076	0,770
Detektor vyšší úrovně	78	14	9	17	38	0,269	0,654

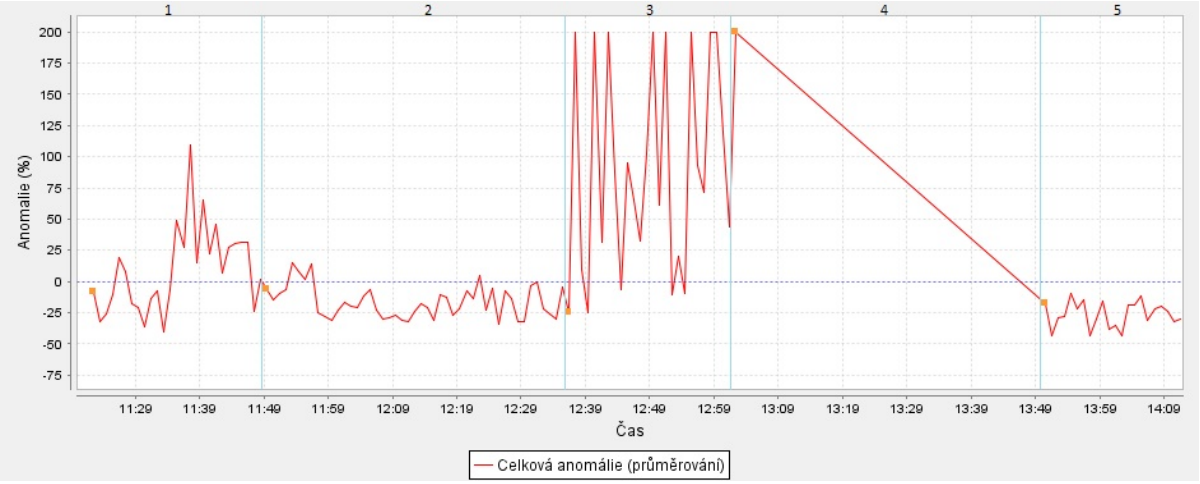
Tabulka 6.18: Vyhodnocení přesnosti detekce

6.8.1.4 Test dat systému zasaženého malware

Poslední ze série testů detekčních schopností HIDS aplikace byl zaměřen na schopnost detekovat anomální chování v případě nakažení počítačového systému škodlivým softwarem. Testovací systém byl před průběhem samotného testu vyčištěn od veškerých nežádoucích aplikací pomocí antivirového a antimalware systému.

V průběhu testu byl testovaný systém bez zabezpečení pomocí firewallu a antiviru vystaven nebezpečnému software, pocházejícímu z návštěvy potenciálně nebezpečných stránek. Po návštěvě těchto stránek byl testovaný systém ponechán bez aktivity. Podezření, že dojde k zasažení malware a jeho aktivaci, se potvrdilo.

V následujícím grafu je zobrazen průběh testu a v tabulce popsány aktivity, které během testu probíhaly.



Obrázek 6.34: Graf vzdáleností vstupních hodnot od prahu (průměrování)

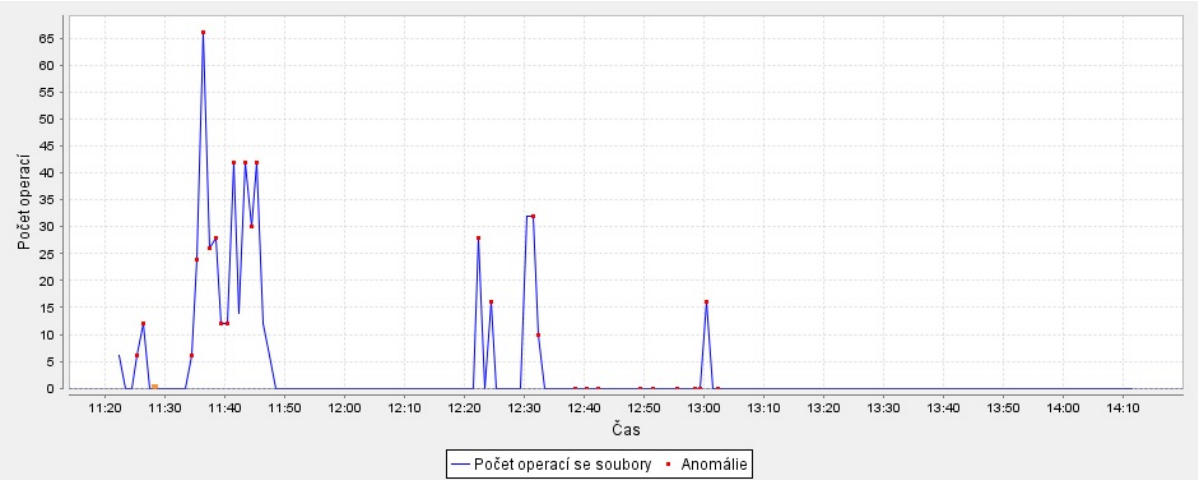
Aktivity vizualizované grafem jsou popsány následující tabulkou. Popis, zda v dané oblasti probíhala normální či anomální aktivita, zde není uváděn, neboť většina činností probíhala skrytě na pozadí běhu operačního systému a o předpokládaných hodnotách anomality by se bylo možno jen dohadovat.

č.	Čas		Popis
	start	konec	
1	11:23	11:49	Umístění klienta do DMZ a načtení několika stránek s možným výskytem potencionálně škodlivého kódu.
2	11:50	12:36	Žádná viditelná aktivita systému, pravděpodobně dochází k infikaci systému.
3	12:37	13:02	Detekovány viditelně anomálie v naměřených datech, škodlivý software byl aktivován a šíří se sítí. V rámci klientské stanice neprobíhala žádná uživatelem viditelná aktivita.
4	13:03	13:50	Klient odpojen od sítě, čištění systému pomocí antivirového a antispywarového software.
5	13:51	14:12	Škodlivý software odstraněn, restart počítače, po kterém nebyla prováděna žádná aktivita.

Tabulka 6.19: Časový průběh aktivit během čtvrtého testu detektoru

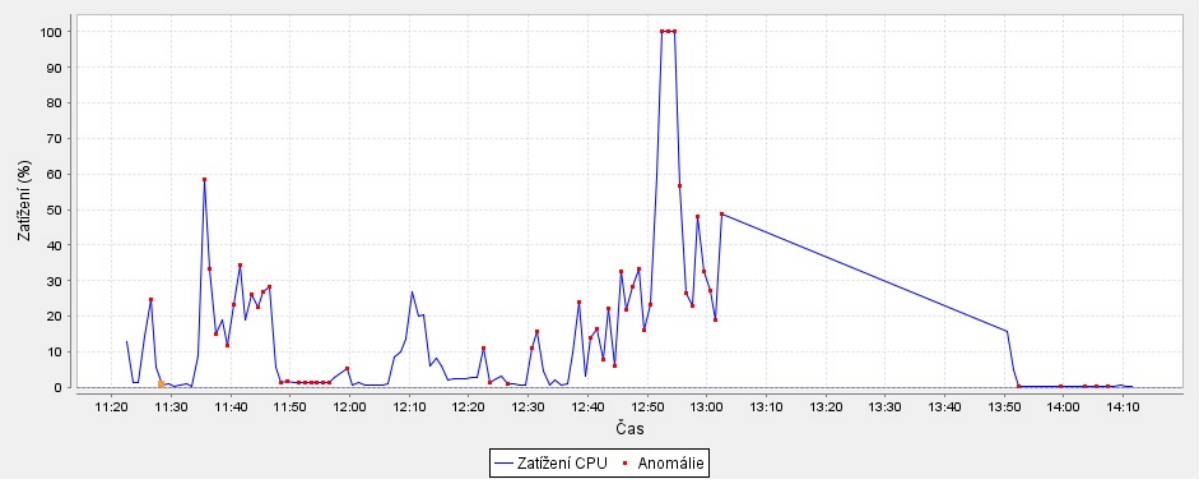
Z grafu je jasně viditelná velká vzdálenost analyzovaných dat od prahu v časovém úseku, v němž byl škodlivý software aktivní, a proto lze detekční schopnosti metody průměrování hodnotit pro účely varování při napadení škodlivým softwarem pozitivně, neboť její výsledky byly v tomto testu zřetelné a jednoznačné.

Běh systému během činnosti nežádoucího software nepůsobil z pohledu uživatele abnormálním dojmem a choval se pro něj standardním způsobem, ovšem při bližší analýze je vidět zvýšené množství operací se soubory v časovém úseku, kdy došlo k návštěvě nebezpečných stránek a nejspíše i k instalaci malware.



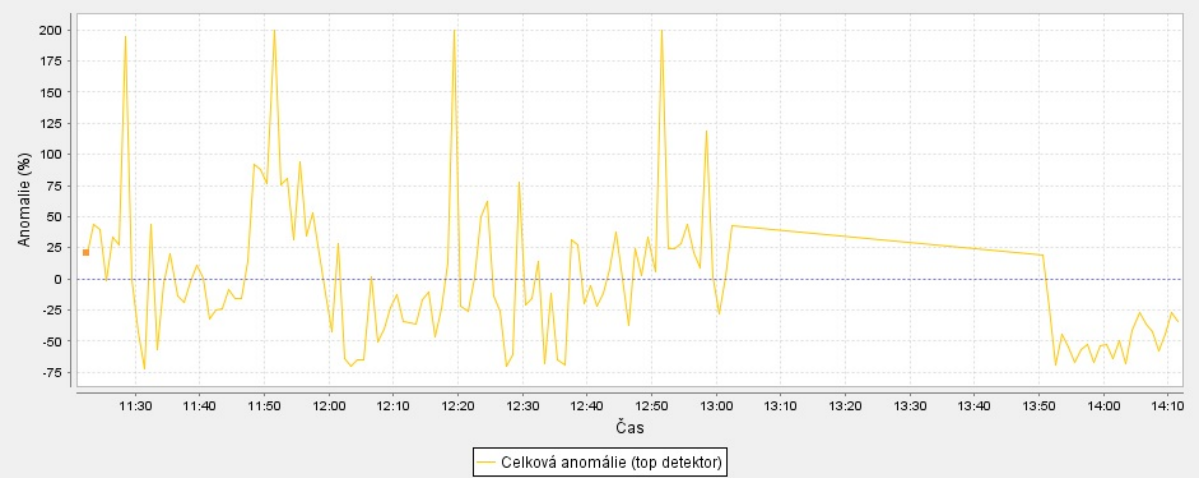
Obrázek 6.35: Graf počtu vykonaných operací se soubory

Druhým výrazně zvýšeným ukazatelem bylo zatížení procesoru v době, kdy byl nežádoucí software aktivní.



Obrázek 6.36: Graf percentuelního využití procesoru

Srovnání metody, která používá pro získání celkové anomálie výpočtu průměru výsledných hodnot jednotlivých detektorů vůči metodě používající detektor vyšší úrovně, nedopadlo pro metodu detektoru vyšší úrovně pozitivně. Z grafu je patrné, že touto metodou nebylo možno útok jednoznačně detekovat.



Obrázek 6.37: Graf vzdáleností vstupních hodnot od prahu (detektor vyšší úrovně)

Pro úplnost je v následující tabulce uveden seznam škodlivého software, který byl po testu odstraněn:

Název	Typ
Trojan Agent/Gen.Process	Trojský kůň
Win32/Sality.NAR	Polymorfní souborový virus
Adware.Vundo/Variant	Spyware
Trojan.Dropper/Sys-NV.Process	Trojský kůň

Tabulka 6.20: Přehled detekovaného škodlivého software

6.8.2 Test výkonnosti HIDS aplikace

Vytvořená demonstrační HIDS aplikace byla testována z pohledu výkonu při detekci anomálií v datech dle naučeného modelu a při profilování chování.

Pro testování výkonnosti HIDS aplikace byl použit osobní počítač s konfigurací, jež je popsána v tabulce 6.21.

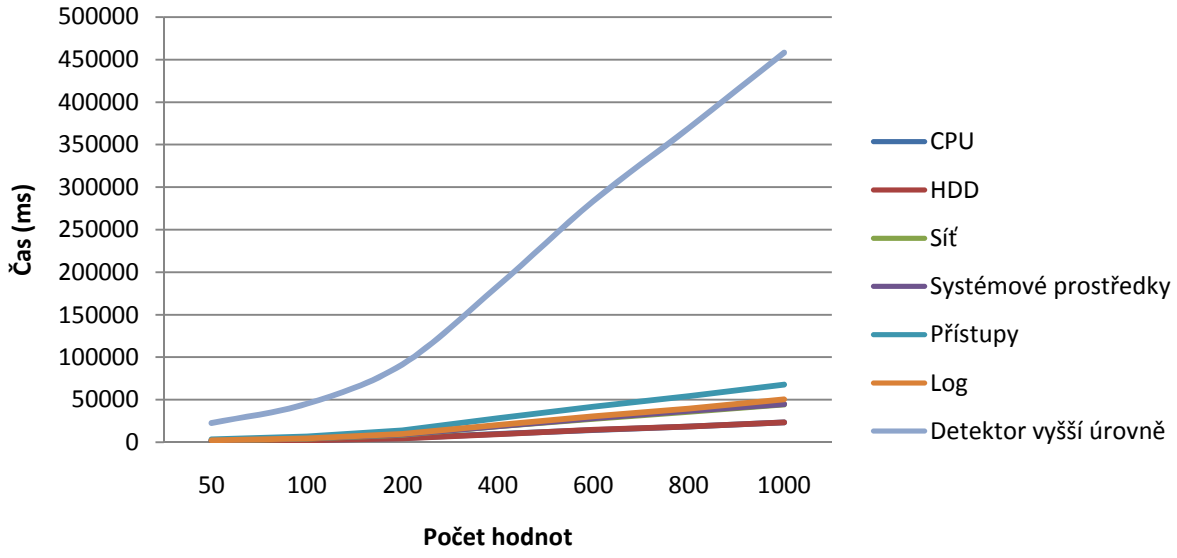
Parametr	Popis
Procesor	Intel Core 2 Quad 6600 @ 2,4 GHz
RAM	4GB DDR2 @ 1066MHz
Základní deska	GIGABYTE EP45-UD3LR
HDD	SEAGATE Barracuda ES.2 1000GB
Operační systém	Microsoft Windows 7 Professional 64bit

Tabulka 6.21: Konfigurace testovacího PC

Test byl prováděn nad několika různě velkými skupinami vstupních dat a dosažené parametry jsou zobrazené v tabulce 6.22 a formou grafu na obrázku 6.38.

Počet hodnot	Počet iterací	Celkový počet průchodů	Čas (ms)						
			CPU	HDD	Síť	Systémové prostředky	Přístupy	Log	Detektor vyšší úrovně
50	500	25000	1162	1202	2206	2280	3396	2484	22752
100	500	50000	2280	2438	4542	4448	6680	4889	45443
200	500	100000	4798	4704	9087	9134	13922	9822	91649
400	500	200000	9688	9551	19261	18778	28359	20384	183729
600	500	300000	14262	14753	27761	28113	42043	30669	283396
800	500	400000	18489	18385	35853	36827	54482	39481	369242
1000	500	500000	23463	23415	44687	44751	67877	50495	457900

Tabulka 6.22: Čas zpracování dat při vytváření profilu

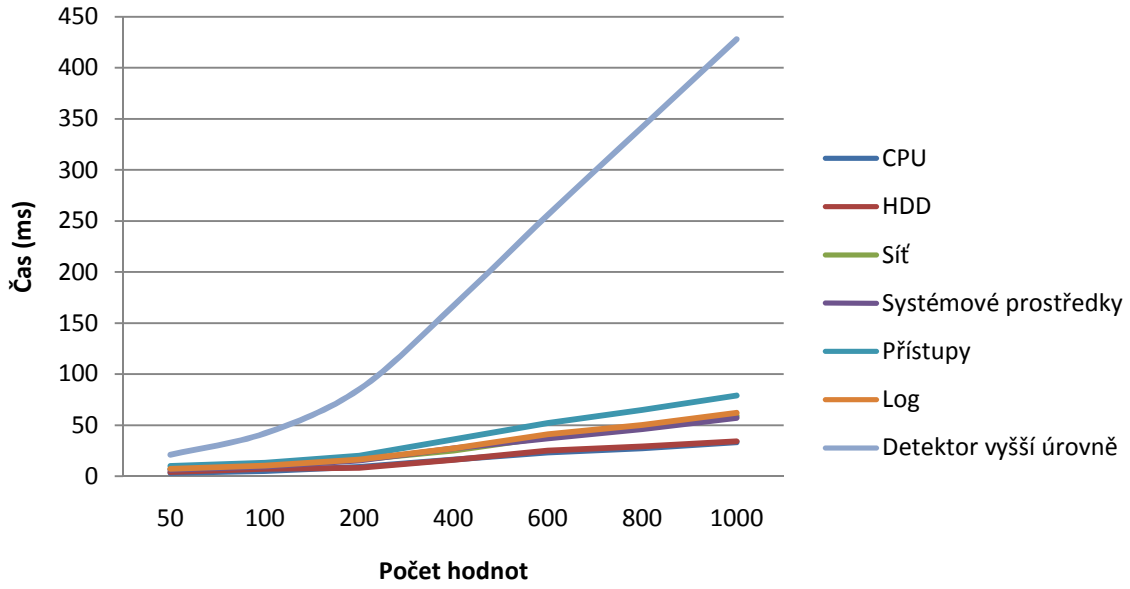


Obrázek 6.38: Časový průběh zpracování dat při vytváření profilu

Dosažené hodnoty při testování výkonnosti demonstrační aplikace v režimu detekce anomálního chování jsou zaznamenány v tabulce 6.23 a formou grafu pak na obrázku 6.39.

Počet hodnot	Čas (ms)						
	CPU	HDD	Síť	Systémové prostředky	Přístupy	Log	Detektor vyšší úrovně
50	3	4	7	6	10	7	21
100	5	7	10	9	13	10	42
200	9	8	16	15	20	16	85
400	16	16	25	27	36	27	167
600	23	25	38	37	52	41	256
800	27	29	47	46	65	50	342
1000	33	34	59	57	79	62	428

Tabulka 6.23: Čas zpracování dat v režimu detekce



Obrázek 6.39: Časový průběh zpracování dat v režimu detekce

Test výkonu detekční aplikace potvrdil předpoklad, že vytváření nového profilu chování je z hlediska času velmi náročnou operací, neboť trénink musí probíhat v mnoha iteracích. Naproti tomu režim detekce je schopen zpracovávat data velmi rychle a systém je schopen zajistit obsluhu většího množství připojených klientů v reálném čase. Optimalizace algoritmu učení tak je vhodným námětem pro budoucí vývoj aplikace.

7 Závěr

Tato diplomová práce se zabývala metodami a možnostmi detekce útoků a anomálního chování. Ve své první polovině popisovala problematiku IDS detekčních systémů a principy jejich činnosti, dále byla věnována pozornost metodám pro detekci anomálií v datech a určení vhodných ukazatelů, nezbytných pro úspěšnou detekci. Druhá polovina práce byla zaměřena na popis návrhu a implementace demonstrační HIDS aplikace. Velká pozornost byla věnována popisu principů činnosti implementovaného detektoru anomálií, založeného na Kohonenových samoorganizačních mapách. Závěr praktické části práce se pak zaměřil na testování vytvořeného řešení, kdy byla provedena série testů se záměrem ověřit detekční schopnosti a výkonnost aplikace.

Na základě experimentů s výslednou aplikací lze říci, že projekt splnil očekávání takřka ve všech bodech. Provedením testů byla ověřena možnost detekce anomálií v datech s využitím Kohonenových samoorganizačních map a prokázal se relativně dobrý potenciál detekčních schopností aplikace, kdy byla identifikována naprostá většina situací, které neodpovídaly normálnímu chování uživatele či aplikací. Navržená aplikace byla schopna vytvořit profily pro většinu činností prováděných uživateli a aplikacemi. V rámci testování bylo experimentováno se dvěma přístupy k určení celkové anomálie vzorku dat a obě tyto metody dokázaly anomálie s větší či menší přesností detekovat, ačkoliv v této oblasti je možno najít prostor pro další vylepšování v budoucnosti.

Z pohledu dalšího vývoje je možno považovat vytvořenou HIDS aplikaci za základ pro její budoucí rozšíření. Jako základní rozšíření stávající aplikace lze uvažovat vytvoření klientských aplikací pro jiné operační systémy než Microsoft Windows. Tento krok by přispěl k širšímu využití aplikace, přičemž je snadno proveditelný díky jednoduše navrženému aplikačnímu protokolu. Mezi rozšíření, která by mohla být v budoucnu navržena a implementována, se řadí také změna struktury detektoru tak, aby bylo možno přesně detekovat ukazatel, který vykazuje anomální hodnoty a nejen skupinu vlastností, kde se anomálie vyskytla. Toto rozšíření by mohlo být realizováno zvýšením počtu Kohonenových samoorganizačních map, sloužících k detekci, až na úroveň, kdy by pro každý parametr existovala jedna samostatná mapa. Pro tento přístup by musel být zřejmě navržen nový mechanizmus, který by vyhodnocoval celkovou anomalitu vzorků, kdy se jako vhodný přístup jeví použití hierarchické struktury Kohonenových map o několika úrovních nebo použití klasifikátoru AdaBoost, který by mohl být ideálním řešením tohoto problému. Dalším vhodným rozšířením by mohla být implementace mechanizmu, který by umožňoval na základě detekovaných anomálií automatické určení typu aktivity, která probíhala. Toto rozšíření by bylo komplexnějšího charakteru a bylo by nutné navrhnout a do aplikace začlenit detektor, umožňující identifikaci na základě rozpoznávání vzorků. Dalšími vhodnými kroky v budoucnu by byla optimalizace detekčního algoritmu pro zvýšení výkonu při vytváření modelů i při detekci anomálií a také snaha o co nejvyšší efektivost detekce.

Literatura

[1] Lundin, N., E. Jonsson. *Survey of Intrusion Detection Research.* Taiwan: Chalmers International Journal of Network Security, 2005.

[2] R. Bragg, M. R. Ousley, K. Strassberg. *Network Security: The Complete Reference.* Columbus: The McGraw-Hill Companies, 2004. 854 s. ISBN 0072226978.

[3] E. Maiwald. *Network security: a beginner's guide*. Columbus: The McGraw-Hill Companies, 2009. 496 s. ISBN 0072229578.

[4] R. Bace, P. Mell. *Intrusion Detection Systems.* In: *NIST Special Publication*. Washington: National Institute of Standards and Technology 2003.

[5] M. Sweeney, C. T. Baumrucker, J. D. Burton. *Cisco security professional's guide to secure intrusion detection system*. Burlington: Syngress Publishing, 2003. 645 s. ISBN 1932266690.

[6] *Application-Based Intrusion Detection Systems*. Washington: National Institute of Standards and Technology, 2003.

[7] S. M. Bechard, S. I. Bechard , A. K. Jones, R. S. Sielken. *Computer system intrusion detection: a Survey.* Charlottesville: University of Virginia Computer Science Department, 2000.

[8] K. Sandeep. *Classification and detection of computer intrusions*. West Lafayette: Purdue University, 1995. Disertační práce.

[9] R. Shimonski, W Schmied. *Building DMZs for enterprise networks.* Burlington: Syngress Publishing, 2003. 774 s. ISBN 1931836884.

[10] D. Duncombe, G. Mohay, A. Clark. *Auto-correlation and Dynamic Attack Redirection in an Immunologically-inspired IDS.* Hobart: ACSW Frontiers '06: Proceedings of the 2006 Australasian workshops on Grid computing and e-research, 2006.

[11] T. Kohlenberg. *Snort Intrusion Detection and Prevention Toolkit*. Burlington: Syngress Publishing, 2007. 750 s. ISBN 1597490997.

[12] *Samhain Labs* [online], c2006 [cit. 2009-12-15]. Dostupný z WWW: <http://www.la-samhna.de>.

[13] *ELM Log Manager : Event Log Management Features* [online]. c2009 [cit. 2009-12-15]. Dostupný z WWW: <http://www.tntsoftware.com/products/ELMLogManager.aspx>.

[14] *PreludeIDS* [online]. c2005-2009 [cit. 2009-12-14]. Dostupný z WWW: <http://www.prelude-ids.com>.

[15] *AlienVault : The OSS Correlation and Security Suite* [online]. c2009 [cit. 2009-12-16]. Dostupný z WWW: <http://www.alienvault.com>.

[16] O. Holub. *Nástroj pro monitorování systémových prostředků OS Windows.* Brno: Vysoké učení technické v Brně: Fakulta informačních technologií, 2008. Bakalářská práce. 40 s.

[17] *Microsoft Developer Network: MSDN Library* [online], 2009 [cit. 2008-04-04]. Dostupný z WWW: <http://msdn.microsoft.com >.

[18] E. Wilson. *Microsoft Windows Scripting with WMI*. Microsoft Press, 2005. 400 s. ISBN: 9780735622319.

[19] P. Malina. *PowerShell – Podrobný průvodce skriptováním*. Brno: Computer Press, 2007. 344 s. ISBN: 9788025118160.

[20] A. Sundaram. *An Introduction to Intrusion Detection.* Crossroads: The ACM Student Magazine, 1996.

[21] A. Veselý, D. Brechlerová. *Neural networks in intrusion detection systems*. Praha: Journal of Forest Science, 2003.

[22] S. Mukkamala, G. Janoski, A. Sung. *Intrusion Detection Using Support Vector Machines.* Heidelberg: The VLDB Journal, 2001.

[23] B. Wotring. *Host Integrity Monitoring Using Osiris and Samhain*. Burlington: Syngress Publishing, 2005. 450 s. ISBN: 9781597490184.

[24] R. J. Hontanón. *Linux praktická bezpečnost*. Praha: Grada, 2003. 440 s. ISBN 8024706520.

[25] L. Vokorokos, A. Baláž, M. Chovanec. *Intrusion Detection System Using Self Organizing Map*. Košice: Acta Electrotechnica et Informatica 2006, 2006.

[26] W. Wang, X. Guan, X. Zhang, L. Yang. *Profiling program behavior for anomaly intrusion detection based on the transaction and frequency property of computer audit data*. Computers & Security, 2006.

[27] H. G. Kayacik, A. N. Zincir-Heywood, M. I. Heywood. *On the Capability of an SOM based Intrusion Detection System*. New York: Engineering Applications of Artificial Intelligence, 2007.

[28] F. González, D. Dasgupta. *Neuro-Immune and Self/Organizing Map Approaches to Anomaly Detection: A Comparison*. Canterbury: In Proceedings of the 1st International Conference on Artificial Immune Systems, 2002.

Seznam příloh

Příloha 1: Obsah přiloženého CD

Přiložené CD obsahuje:

- Zdrojový text této práce v Microsoft Word formátu v adresáři: „**.\Technická zpráva**“
- Text této práce v PDF formátu v adresáři: „**.\Technická zpráva**“
- Demonstrační HIDS aplikaci
    - Kompletní zdrojové kódy včetně všech potřebných knihoven v adresáři: „**.\Demonstrační aplikace\Zdrojové kódy**“
    - Spustitelnou verzi aplikace v adresáři: „**.\Demonstrační aplikace\Spustitelná verze**“
- Demonstrační data v adresáři: „**.\Demonstrační aplikace\Demonstrační data**“
- Soubor **readme.txt** s přesným popisem struktury CD